UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

TÉCNICO LISBOA

Energy-aware Service Provisioning

September 2016

Energy-aware Service Provisioning in P2P-assisted Cloud Ecosystems

Leila Sharifi

September 2016

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

TÉCNICO LISBOA

Ph.D. Thesis in Computer Architecture

TÉCNICO LISBOA

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

Energy-aware Service Provisioning in P2P-assisted Cloud Ecosystems

Leila Sharifi

PhD Thesis in Computer Architecture
Lisbon, Portugal and Barcelona, Spain
2016

Abstract

Energy has been emerged as a first-class computing resource in modern systems. The trend has primarily led to the strong focus on reducing the energy consumption of data centers, coupled with the growing awareness of the adverse impact on the environment due to data centers. This has led to a strong focus on energy management for server class systems.

In this work, we intend to address the energy-aware service provisioning in P2P-assisted cloud ecosystems, leveraging economics-inspired mechanisms. Toward this goal, we addressed a number of challenges.

To frame an energy aware service provisioning mechanism in the P2P-assisted cloud, first, we need to compare the energy consumption of each individual service in P2P-cloud and data centers. However, in the procedure of decreasing the energy consumption of cloud services, we may be trapped with the performance violation. Therefore, we need to formulate a performance aware energy analysis metric, conceptualized across the service provisioning stack. We leverage this metric to derive energy analysis framework.

Then, we sketch a framework to analyze the energy effectiveness in P2P-cloud and data center platforms to choose the right service platform, according to the performance and energy characteristics. This framework maps energy from the hardware oblivious, top level to the particular hardware setting in the bottom layer of the stack.

Afterwards, we introduce an economics-inspired mechanism to increase the energy effectiveness in the P2P-assisted cloud platform as well as moving toward a greener ICT for ICT for a greener ecosystem.

Keywords— P2P en la nube, centro de datos, la eficiencia energética, el modelado de energía, smart grid.

Resum

La energía se ha convertido en un recurso de computación de primera clase en los sistemas modernos. La tendencia ha dado lugar principalmente a un fuerte enfoque hacia la reducción del consumo de energía de los centros de datos, así como una creciente conciencia sobre los efectos ambientales negativos, producidos por los centros de datos. Esto ha llevado a un fuerte enfoque en la gestión de energía de los sistemas de tipo servidor.

En este trabajo, se pretende hacer frente a la provisión de servicios de bajo consumo energético en los ecosistemas de la nube asistida por P2P, haciendo uso de mecanismos basados en economía. Con este objetivo, hemos abordado una serie de desafíos.

Para instrumentar un mecanismo de servicio de aprovisionamiento de energía consciente en la nube asistida por P2P, en primer lugar, tenemos que comparar el consumo energético de cada servicio en la nube P2P y en los centros de datos. Sin embargo, en el procedimiento de disminuir el consumo de energía de los servicios en la nube, podemos quedar atrapados en el incumplimiento del rendimiento. Por lo tanto, tenemos que formular una métrica, sobre el rendimiento energético, a través de la pila de servicio de aprovisionamiento. Nos aprovechamos de esta métrica para derivar un marco de análisis de energía.

Luego, se esboza un marco para analizar la eficacia energética en la nube asistida por P2P y en la plataforma de centros de datos para elegir la plataforma de servicios adecuada, de acuerdo con las características de rendimiento y energía. Este marco mapea la energía desde el alto nivel independiente del hardware a la configuración de hardware particular en la capa inferior de la pila.

Posteriormente, se introduce un mecanismo basado en economía para aumentar la eficacia energética en la plataforma en la nube asistida por P2P, así como avanzar hacia unas TIC más verdes, para las TIC en un ecosistema más verde.

palabras-clave— nube asistida por P2P, centro de datos, eficiencia energética, modelado de energía, smart grid.

Acknowledgements

The journey toward getting a PhD is full of ups and downs, joys and disappointments, failures and successes. This journey cannot be completed without the amazing people who generously give advices and support to a PhD student.

Foremost, I would like to express my deepest gratitude to my advisor, Dr. Luís Veiga. Your broad knowledge, expertise and experience as well as your incredible enthusiasm, warm encouragements and continuous support and patience provided me with a great opportunity to do this PhD. You positive attitude toward the work had been moving me forward throughout my PhD studies. Without your advices this dissertation would never have been materialized.

I also would like to appreciate my co-advisor Dr. Felix Freitag for his support and providing the required research equipments during my stay at UPC. I would like to offer my cordial thanks to Dr. Llorenç Cerdà-Alabern for our collaboration on QMPSU network energy analysis. It has been an honor working with you and benefiting from your extensive knowledge in the area of networking. You have been enthusiastically open to a discussion all the time and generously allotted a lot of time to share your knowledge with me. My special thanks goes to Dr. Jordi Guitart for the illuminating discussions, and constructive feedback on my work.

I am thankful to Dr. José Simão for the assistance in the work on energy analysis framework. I would like to express my thanks to Navaneeth Rameshan for many fruitful and enthusiastic discussions as well as the collaborations we had during this PhD course.

I am also obliged to my colleagues and fellow students at INESC-ID and UPC, for contributing to such a great and friendly ambiance to do research.

I would like to extend my appreciation to Barcelona Supercomputing Center, for providing the infrastructure for part of my experiments. I acknowledge the financial support I have received from European Commission under EMJD-DC program and DependableCloud project.

A big thanks to my dear friends, without their emotional and friendly support, I could not have explored this PhD course. And most importantly, tons of thanks to my dear family (my great daddy Nouraddin, my lovely, little sister Zahra and in memory of my mom Jila), whose endless love and spiritual support fills me with life.

Above all, to my love Mehdi for his invaluable support, unconditional love and endless patience, I am particularly grateful. You have been my source of inspiration all the time, since I first met you. I dedicate this work to you with all my love and gratitude.

Contents

List of Figures

List of Tables

Part I

Thesis Overview

—"Setting goals is the first step in turning the invisible into visible."

-Tony Robbins

1

Introduction

Cloud computing is becoming the predominant IT service provisioning paradigm, due to its availability, reliability and cost effectiveness; no hardware and infrastructure capital investment is required, in the user side. Nonetheless, energy and associated environmental costs (cooling, carbon footprint, etc.) of IT services constitute a remarkable portion of services' operational expenditure. For instance, cloud computing energy consumption is expected to reach to 1,963 billion kWh by 2020 [1].

Indeed, there is a need for an energy aware cloud architecture and economic model which includes service pricing and resource allocation. Since the energy price is going to dominate other service costs, it is required to devise an energy-based pricing mechanism for each service. To this end, we need to formulate per job

energy consumption estimation techniques in order to schedule resources in an energy efficient manner.

Although there is a growing body of work centered on the field of energy aware resource management, allocation, scheduling and pricing [67, 75, 76, 133], they mainly considered the whole system energy measurement, estimation, improvement and optimization. There is only limited work focusing on the energy issues per job [23, 28, 68, 76]. However, they only aim at reducing total power consumption in the infrastructure without taking into account the energy-related behavior of each individual job, its performance and price, i.e., how expensive and useful is the energy employed for the observed job performance or progress.

Energy-based job pricing confronts some more challenges further to the system wide energy efficiency issues. In the system wide energy efficiency, the energy consumption of the resources are measurable simply by plugging the energy meter devices or exploiting the embedded sensors of the contemporary devices, e.g. Runtime Average Power Limit (RAPL) counters in recent Intel CPUs. Nonetheless, it is nontrivial to measure the energy consumed per job, since we cannot embed a physical sensor in a job or plug a metering device to it. Therefore, estimation is still the only option in this case. Estimation results in a more complicated model since it has to deal with uncertainty and error. Moreover, the need for a general approach that assumes an unpredictable workload behavior aggravates the problem. Besides, energy non-proportional hardware infrastructure adds additional complexity to the agenda in the multi-tenant[1] ecosystems.

By estimating the energy consumption of each job, the attempt is to assign it to the hosts which incur the lowest energy dissipation. To this end, we need an energy aware resource manager which is aware of the power sources. The more diverse the energy providers, the greater the variety of user choices. Smart grids naturally fulfil this goal. Diverse energy sources of smart grid improves the availability, sustainability and environment friendliness of the cloud services. Hence, combining

[1]Note that in this work, we consider multi-tenancy and co-location as the same concept in which the VMs of independent applications from the same user are treated as the VMs of different applications from two different users.

the cloud infrastructure with smart grid can improve the economic model of both systems.

1.1 Problem Formulation

We identify several requirements for an energy-aware economic model for the P2P-assisted cloud systems which are not well developed so far. The energy efficiency of P2P-clouds in comparison with the classic cloud model is still a matter of controversy. Furthermore, energy efficiency has not been a priority concern in the end-user incentive list. This will tend to change in the midterm future as domestic prices of electricity rise and the profiles of energy sources are factored in the prices (e.g. by computing utility providers/distributors).

The main problem to address in this work is to introduce an economic model for the P2P-assisted clouds which is centered on the energy based pricing of the services, that embeds the energy efficiency in user incentive list, and reduces the carbon footprint to be more environment friendly.

A distinguishing point of our model is considering the energy from a consumer perspective, i.e. per job, in lieu of the coarse grain provider vantage point. Our ultimate goal is to increase the energy efficiency through the pricing mechanism. For this purpose, a P2P-assisted cloud, as studied in [113, 117, 118], outperforms the classic data center oriented cloud architecture, due to the diverse range of processing elements scattered all through the system, which can accomplish certain sorts of tasks with lesser energy dissipation. However, there is no general recipe for energy conservativeness of each platform. Therefore, a generic framework which can compare the energy efficiency of different systems is required.

Comparison results feed the service assignment process, which is aiming at lower energy dissipation. Moreover, to this end, we need an energy aware resource manager which is aware of the power sources. The more diverse the energy providers,

the greater the variety of choices. Smart grid provides a diverse range of energy sources including renewable and clean sources. Hence, combining the cloud infrastructure with smart grid directs us to attain lower energy consumption.

1.2 Research Questions

To formulate an economic model for energy aware service provisioning in P2P-assisted cloud ecosystems, we need to justify the tentative energy savings on the integrated platform. To this end, the following question should be addressed. **Q1: Is it energy efficient to switch to P2P-cloud?** We need a framework to analyze the energy consumption of a service across different platforms,and hence, the following question emerges as a consequence. **Q1.1: What are the requirements of the energy consumption analysis framework?** We should note that aiming just at decreasing energy consumption may result in performance plunge. Therefore, a performance aware energy analysis metric is needed in the introduced framework. This metric should be able to chatacterize and assess energy and performance across the service provisioning stack. We address this issue by answering the following question. **Q1.2: Which metric should be applied for the analysis?**

An energy resource aware resource manager is needed to improve the energy conservativeness of the system. Exploiting distributed energy sources contribute to a more sustainable system, due to the resource diversity and facilitates the move toward greener ecosystem, which has been emerging as the prime system requirement. To fulfil this need, we should tackle the following question. **Q2: How can we exploit diverse energy sources?** Combining energy and information systems can be a solution. The integration of the two aspects in the system can contribute to a greener ubiquitous society by equipping it with the concept of energy conservativeness, and leveraging renewable energy sources. Therefore, developing this solution requires to elaborate the following detailed questions. **Q2.1: What are the benefits in the collaboration between distributed energy and**

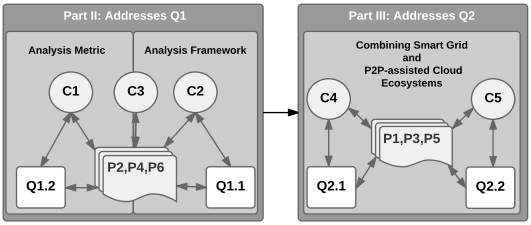


FIGURE 1.1: Work outline

computing resources? Q2.2: Is the collaboration between distributed energy and computing resources feasible?

1.3 Contributions

In this section, we outline the major contributions of the thesis by mapping each contribution to the associated research question. The relation of each question, the contributions and related publications are illustrated in Figure 1.1. The major contributions of the thesis are listed as follow:

C1: A metric for performance-aware energy analysis across the service provisioning stack. We introduce energy effectiveness in **P**4, as an adaptive, combined metric that considers performance and energy simultaneously in quantifying the service efficiency. This contribution specifically addresses Q1.2 and is discussed in Chapter 3.

C2: A framework to analyze energy consumption in P2P-assisted cloud ecosystems. Comparing the energy effectiveness in P2P-cloud and data center requires a framework to analyze energy consumption in each platform. Hence, to answer Q1, and in particular Q1.1, we sketch an energy effectiveness analysis framework in **P**6, which is studied in Chapter 4.

C3: Comparing the energy consumption of P2P-cloud and classic cloud models. This contribution aimed to find an answer for Q1, through analysis and experiments as illustrated in **P**2, **P**4 and **P**6. Detailed discussion appears in Chapter 4.

C4: Study the viability of smart micro-grid and P2P-cloud integration.
A tentative answer for the second question, specially Q2.1, is the combination of smart grid and cloud systems to make the cloud energy resources aware. We study the viability of this integration in Chapter 5. Relevant publication to this contribution is **P**1.

C5: A framework to integrate smart grid and cloud service provisioning.
To answer Q2.2, we introduce Cloud of Energy (CoE), as an integration framework for smart grid and P2P-assisted clouds. This framework enables energy-aware service provisioning for computing services through an economic middleware. CoE is introduced in **P**3 and **P**5. Detailed study of CoE is available in Chapter 5.

1.4 Relevant Publications

P1. Sharifi, Leila, Felix Freitag, and Luis Veiga. *"Combing Smart Grid with community clouds: Next generation integrated service platform."* In IEEE Smart Grid Communications (IEEE SmartGridComm), 2014 [114].

In this paper, we compare smart micro-grid and P2P-cloud and qualitatively elaborate their collaboration potential. As a case study, the benefits of leveraging P2P-cloud as the computing and communication platform in micro-grid level is studied.

P2. Sharifi, Leila, Navaneeth Rameshan, Felix Freitag, and Luis Veiga. *"Energy Efficiency Dilemma: P2P-cloud vs. data center."* In IEEE CloudCom2014, **Best Paper Candidate** [117].

In this paper, we investigate the energy consumption of different cloud architectures, from a mega-data center to a P2P-cloud that provides extreme decentralization in terms of data center size. Our evaluation results reveal the fact that the more decentralized the system is, the less energy may be consumed in the system.

P3. Sharifi, Leila, Felix Freitag, and Luis Veiga. *"Envisioning Cloud of Energy."* In IEEE Smart Grid Communications (IEEE SmartGridComm), 2015, **Best video award** [115].

In this paper, we outline the idea of Cloud of Energy (CoE) which fosters the adoption of green energy and green cloud by integrating these two systems. CoE introduces an integrated framework of everything as a service to facilitate the service exchange, not only across the computing and electricity grid hierarchy, but also among them via an economic middleware.

P4. Sharifi, Leila, Llorence Cerda-Alabern, Felix Freitag, and Luís Veiga. *"Energy Efficient Cloud Service Provisioning: Keeping Data Center Granularity in Perspective."* Journal of Grid Computing, Springer, June 2016, Vol. 14, Issue 2, pp 299-325 [113].

In this paper, we elaborate on the energy effectiveness of service provisioning on different cloud architectures, from a mega-data center to a nano-data center, which provides the extreme decentralization in terms of cloud architecture, as well as P2P-clouds or community network clouds. We study the energy consumption through an analytical and simulation framework for video streaming and MapReduce applications.

P5. Sharifi, Leila, Felix Freitag, and Luis Veiga. *"ARTA: An Economic Middleware to Exchange Pervasive Energy and Computing Resources."* To appear in IEEE Smart Grid Communications (IEEE SmartGridComm), 2016, [116].

In this paper, we introduce an agent-oriented economic middleware architecture (ARTA) to exchange pervasive energy and computing resources in different layers of the service provisioning platform, from the edge layer of micro-grid and P2P-cloud to the mass production layer of the giant power plants and data centers. ARTA follows a semi-decentralized economic model by operating through partial system view in the edge-layer negotiations and considers system dynamics and uncertainties in the agents decisions.

P6. Sharifi, Leila, Jose Simao, Navaneeth Rameshan, Felix Freitag, and Luis Veiga. *"A Framework to Analyze Energy Effectiveness in P2P Assisted*

Cloud Ecosystems." Submitted to IEEE Transactions on Cloud Computing, **Invited Paper- Under Review** [118].

In this paper, we devise a hardware agnostic, analytical framework to outline the energy effectiveness of current and upcoming cloud paradigms across the stack, from the architecture to the services. In particular, the framework is used to assess infrastructure energy usage considering its energy effectiveness. This assessment resorts to representative workloads in cloud settings, such as storage services, and MapReduce jobs.

1.5 Thesis Road Map

This document is structured in a bottom-up approach, as visualized in Figure 1.1. After outlining the background concepts, in the next chapter, we formulate a performance aware energy analysis metric across service provisioning stack in Chapter 3. Then, in our path toward framing our energy aware service provisioning platform, in Chapter 4, we introduce a framework to compare the energy effectiveness in P2P-cloud with data center. Afterwards, we outline a possible solution to exert distributed renewable energy sources of smart grid, as well as distributed processing elements of P2P-cloud assisting mass producers to achieve a greener ecosystem in Chapter 5. The work is concluded in Chapter 6.

—"*Once we accept our limits, we go beyond them.*"

-Albert Einstein

2

Background Concepts

In this chapter, we define energy efficiency and its synonyms as the foundation of our work. Then, we outline the elaborated cloud models, terms and hypothesis employed all through this document. We initially address the relevant cloud platforms; then, we address virtualization in the context of data centers as well as in peer-to-peer deployments. In the latter part, we introduce the smart grid concept, which plays a role in the final framework that we sketch in Chapter 5.

2.1 Energy Efficiency

Since the energy efficiency has been introduced , different approaches toward this concept has been formed. Energy efficiency in the literature is interpreted either as energy consumption saving, energy cost saving or energy sustainability [51].

Energy cost represents the money required to produce, transmit, and consume energy. Therefore, energy cost saving policies are oriented toward reducing the mentioned monetary cost [51]. Since the cost of powering the hardware outweighs the commodity hardware price, energy cost saving is becoming imperative in cloud computing.

On the other hand, energy sustainability refers to the development of a sustainable system regarding energy to meet the present needs, without compromising the future generation's needs [119]. Sustainability in cloud computing is majorly implemented through green computing and infrastructure design techniques [78].

Green computing is a subclass of sustainable computing. It aims to reduce the use of hazardous materials, maximize energy efficiency during the product's lifetime, and promote the recyclability or biodegradability of defunct products and factory waste. Green computing is important for all classes of systems, ranging from handheld systems to large-scale data centers.

In the rest of this text, we focus on energy consumption saving which addresses the challenge of reducing the energy consumption. We use energy cost saving (referred as energy saving for brevity), energy capping, and energy conservativeness terms interchangeably throughout the text.

Moreover, energy related studies in computing can be classified as hardware level, virtualization layer and software/application level [18]. While hardware level studies basically include design and implementation of energy efficient hardware, VM level and software level studies focus on exploiting the hardware in an energy efficient way by introducing energy efficient resource allocation mechanisms. Considering sustainability, resource allocation is defined as dynamic load balancing to promote renewable energy sources and renewable based power capping policies [119].

2.2 Cloud Platforms

In this section, we study different cloud platforms introduced so far.

2.2.1 Classic data center

In the classic data center model, as the base of the cloud computing concept, a gigantic data center embraces a number of host clusters, constituting a powerful computing or storage capacity. The internal organization and hierarchy of the data center can follow a number of variants, as we discuss in Section 2.3, typically aiming to reduce latency and energy consumption in internal traffic.

2.2.2 Distributed data center

Distributed cloud architectures consist of a large number of small-sized data centers distributed across diverse geographic locations. This architecture is appealing for network service providers who already have the necessary distributed facilities such as distributed offices that are geographically dispersed to be closer to users, since they can develop a large number of distributed data centers interconnected by high-speed networks.

Distributed cloud architectures [6] can provide several benefits over the traditional centralized cloud architectures. With distributed data centers, requests can be served locally, this helps reduce network capacity needs, for high-bandwidth applications, which constitute a significant cost when accessing centralized data centers. Distributing the data centers also reduces the latency of access compared to traditional data centers; distribution of data centers has long become common with content delivery networks [43]. In fact, the access latency of the traditional data center may have large variations due to the longer path lengths and the need for frequent switching among multiple service providers.

What is more, the challenges of supporting business continuity in a cloud environment are not limited to a physical area or data center. The elasticity and flexibility of the network architecture must be addressed as well. Therefore, the computing, storage, and network components used for cloud computing should not all reside in the same physical location. These resources could be spread over multiple locations and interconnected using a transparent transport mechanism that maintains security and end-to-end segmentation. A Distributed cloud data center, in addition to bringing high availability and disaster recovery, provides the opportunity to use different, local energy sources.

2.2.3 Cloud Federation

The cloud is mostly about the elasticity and flexibility and the network architecture is a key component that helps drive these properties. Federated Cloud features the same architecture as the distributed data center. The only difference is in providing the resources through the aggregation of several providers in the federation, while all the infrastructure remains under the control of a single provider in the distributed data center model. Thus, regarding topology, federated clouds are distributed data centers, only subject to different administrative and organizational domains.

2.2.4 Nano data center

To take further advantage of distribution, the concept of the nano data center(NaDa) [129] takes to the extreme the distribution already present in the concept of distributed data center architecture. In a nano data center, each host stands for a data center that connects to other nano data centers via the Internet. The advantage of nano data center model is that it is exempted of any relevant effort for cooling or maintaining the data center. Moreover, if location-aware resource allocation mechanisms are applied, the nano data center model performs better than the classic data center model, thanks to the distributed architecture.

NaDa's goal is to combine the power of data centres with the scalability of peer-to-peer (P2P) methods, while not threatening the robustness and the stability of the cloud services. While most research worldwide is focused on increasing the efficiency of network architectures deployed around the network backbone and data centres, NaDa is taking a radically different approach, in which network is untouched and the content is located on customer premises.

To combine all unused edge resources, NaDa attempts to use a new, managed P2P communication architecture. The P2P paradigm aids in the deployment of new services such as file sharing or IP-based telephony because it avoids having to scale servers for peak capacity. However, most of the currently deployed P2P systems have been focused on simple file sharing or streaming applications (and often for illegal content). Thus, several fundamental issues must still be addressed in order to reinvent the P2P paradigm for the NaDa system.

2.2.5 Micro-cloud

The micro-cloud concept is introduced by the Supermicro company [1]. Supermicro offers a micro-cloud system as high-density, rack-mounted server units each populated separately with CPUs, memory, and SSD drives. These compact racks can be connected on a rack and can extend the computing power of the cloud with less complexity than that of the classic data canter infrastructure. Micro-cloud items can be exploited in server centric data center networks, such as BCube [56] to build a modular data center; Bcube is explained in Section 2.3.

Modular data centers have some benefits compared with the data center facilities directly built from server racks: these include shorter deployment time, higher system and power density and lower cooling and manufacturing cost. However, it is difficult or even impossible to service a modular data center once it is deployed.

[1] `https://www.supermicro.com/white_paper/white_paper_DCO.pdf`

2.2.6 Edge(Fog) computing

The term “fog computing” [29] is introduced by Cisco Systems as a new paradigm to support wireless data transfer via distributed devices in the “Internet of Things(IoT).” Its hierarchical distributed architecture extends from the edge of the network to the core in order to provide a geo-distributed platform that improves location awareness and latency. These features suit the IoT platform best, because the big is in the number of data sources rather than the data volume in IoT.

Edge (fog) Computing is pushing the frontier of computing applications, data, and services away from centralized nodes to the logical extremes of a network. It enables analysis at the edges (i.e. at the source of the data). Thus, services would be located closer to the end-user to improve on latency and data access compared with those of the data center model.

However, fog computing is not a substitute for the cloud, it is rather a complement. There is a fruitful interplay between cloud and fog, particularly when it comes to big data management and analytics. By controlling data at various edge points, fog computing integrates core cloud services with those of a truly distributed data center platform. With fog services, we are able to enhance the cloud experience by isolating user data that is required to live on the edge. This infrastructure still maintains the concept of the cloud while incorporating the power of fog computing at the edge.

Fog computing extends direct cloud services by creating an edge network which sits at numerous points. This dense, geographically dispersed infrastructure, helps in numerous ways. Big data and analytics can be performed faster with better results. Administrators are able to support location-based mobility demands and not have to traverse the entire Wide-area Network (WAN), in this way it provides true support for mobility and the IoT. Not only does fog computing improve user perceived performance (latency, throughput), it also helps with security and privacy issues. These edge (fog) systems would be created in such a way that real-time data analytics become a reality on a truly massive scale.

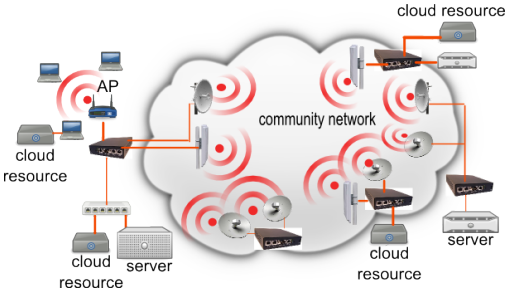


FIGURE 2.1: P2P-cloud intra-vicinity model

Fog computing is applicable in geo-distributed applications with very low and predictable latency [29] (e.g. pipeline monitoring application, mobile applications and large scale distributed control systems, e.g. smart grids). Namely, fog computing can be leveraged as the smart micro-grid level distributed data processing [114], and can contribute to bi-level data aggregation in P2P-assisted cloud platforms [115].

2.2.7 P2P-cloud

The concept of the P2P-cloud is the extended idea of nano data center that replaces the expensive server machines with commodity hosts that are combined with the edge computing platform. A P2P-cloud embraces a set of commodity hosts, including IoT boards, laptops and PCs, connected via a wireless communication platform as depicted in Figure 2.1. The main goal of the P2P-cloud is to take advantage of distributed data center hosts as well as exploit the commodity hardware of community networks [32]. Hence, they are also often described in the literature as *community network clouds* or *community clouds*, although this last notion should not be confused with NIST [101] definition of community clouds, that addresses closed multi-tenant infrastructures hired or owned by different tenants with shared concerns (e.g., mission, security, compliance) to reduce overall costs due to sharing.

The P2P-cloud topology we address here is the vision of a cloud deployment in community networks: a cloud hosted on community-owned computing and communication resources providing a diverse range of services. Community networks represent the collaborative effort of community members, for building an ICT infrastructure with commodity devices in a bottom-up approach, in order to meet demand for the Internet access and services [74].

The P2P-cloud benefits from the geo-distributed infrastructure such as nano data centers and edge computing platforms, which contributes to location awareness and reduced communication latency for the locally provided services. However, it differs from other edge computing efforts, because it actually shares the available resources among users.

Comparing P2P-cloud with desktop grid [36], we discover that desktop grid is a peer-to-peer volunteer computing platform. However, P2P-cloud services are not confined simply to computing. Moreover, the concept of P2P-cloud may be mixed up with mobile cloud or cloud offloading. Namely, P2P-cloud is a broad concept that embraces all the above mentioned concepts. To exemplify, P2P-cloud hosts may be mobile or static. P2P-cloud reinforces the concept of telco-cloud or telco-backed cloud [147], because communication and IT infrastructures akin to the community network are required to develop a P2P-cloud.

In a P2P-cloud, energy is substantially consumed at hosts, switches, routers and network devices. Compared to the classic clouds, in communities, we encounter much reduced static energy waste, because the machines which do not serve the community may already be on to serve the users' individual applications. Moreover, the Idle to Peak power Ratio (IPR) for the current P2P-cloud hosts is close to the ideal case, and the PC machines consume slightly less energy compared to the data center servers.

What is more, in P2P-cloud, to alleviate the energy consumption, requests can be assigned to the closest available host in the community. According to this fact, we introduce the P2P-cloud topology as a set of community hosts scattered within

vicinities and communicating via the wireless communication network as depicted in Figure 2.1. Each vicinity can access the others via the Internet.

2.2.8 P2P-assisted Cloud

Federation of P2P-cloud and data centers through the concept of the fog, elevates the popularity of cloud systems due to the advantages of reduced latency, higher availability and cheaper services and better quality of service.

Service prices can be reduced by pushing the computing toward the commodity devices at the edge of the network; however, data centers still support the services in the backbone in case of failure or if a service demands specific computing requirements which can be better provided via the data center servers, e.g. parallel data processing in specific MapReduce scenarios [117]. In interactive applications, P2P-cloud platform can decrease the latency compared to the data centers by local service provisioning in a geo-distributed platform.

This P2P-assisted cloud model suits the locality of services more than classic clouds. Loosely paraphrasing, in this model, each host is adaptable to a specific architecture, configuration and service according to the most prevalent requests it receives. This idea is inspired from the peer-to-peer content and location aware overlay construction [70, 82, 84].

Studies reveal that virtually all the requests a user issue for the service, in a specific location, are similar to the others due to the locality of requests [63]. The P2P-cloud can leverage this fact by adjusting the service and computing capabilities of each individual community nodes accordingly, by responding to high resource demanding requests via the federation on more powerful machines in classic clouds, and form the P2P-assisted cloud.

2.3 Intra-data center topology

With increasing computing power in data centers, I/O (a significant part of it in networking) becomes a bottleneck against performance. Conversly, as servers become more energy proportional, the data center network dissipates a remarkable fraction of cluster power. The energy consumption of different structures of data centers with emphasis on energy efficiency of data center architecture has been studied in [25, 61].

Data center topologies, typically, are classified into two categories of switch centric and server centric architectures. The efforts in the design of electronic data centers have succeeded to mitigate many dilemmas and obstacles in providing switch centric architectures which significantly rely on the expensive high-performance switches that can support fault tolerance, load balancing, and agility, and are able to handle high oversubscription ratios. Switch-centric category includes conventional tree topology, Fat-tree [5], VL2 [53], QFabric [3], etc.

Alternatively, server centric data centers, on the other hand, use servers as relay nodes, and provide an infrastructure with low diameter and high capacity, while employing commodity switches in order to support different traffic types for applications with intensive computing requirements. Nevertheless, additional wiring cost and complexity are a result of having servers equipped with more than one port. Server centric topology includes BCube [56], Pcube [65], CamCube [41], etc., as explained in the rest of this section.

2.3.1 Three-tier architecture

As depicted in Figure 2.2, the classic network architecture consists of three layers of core, distribution and access. Core layer infrastructure and topology are designed to fulfill the high-performance forwarding, while distribution and access layer offer lesser performance and more complicated topologies, but only require commodity hardware.

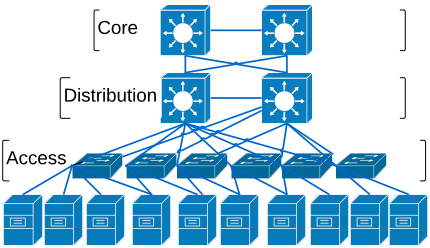


FIGURE 2.2: Intra-data center hierarchical communication model

2.3.2 Fat-tree

This topology [5], consists of servers, edge, aggregate and core switches. A set of edge and aggregate switches are connected as a Clos topology and form a complete bipartite in each pod. A Clos network is kind of a multistage circuit switching network, which represents a theoretical idealization of the practical multi-stage telephone switching systems [38]. Each pod is connected to all core switches forming another bipartite graph. Fat-tree IP addresses are in the form of 10:pod:subnet:hosted. Applying Fat-tree topology issues with oversubscription, costly aggregation, fault tolerance, and scalability are resolved. Most of the switch centric architectures are based on Fat-tree.

2.3.3 Virtual Layer 2 (VL2)

VL2 is designed based on the Fat-tree topology [53]. However, core and aggregate switches in VL2 implement a Clos topology. VL2 relies on commodity switches since it employs Equal Cost Multi Path (ECMP) for packet forwarding and topology updates. It implements valiant load balancing by randomly selecting an intermediate switch before forwarding a packet and applies flow hashing in ECMP.

2.3.4 Qfabric

This data center network topology provides one-hop paths with no oversubscription and congestion as a result. It is introduced by Juniper [3] to simplify the data center management via reducing the number of switches. Flattening three-tier tree structure to one tier fabric is a solution proposed for modern data center architecture to tackle the oversubscription, costly aggregation, fault tolerance and energy efficiency in the data center network by reducing the number of switches and path length among servers.

These kind of architectures are called energy efficient data center topologies due to the energy saving on both static power, by reducing the number of switches; and dynamic power, by reducing the path length. Flattended-butterfly [4] is following the same goal by flattening the butterfly data center network topology. FlatNet [81] is another double layer flattened topology, that attempts to fulfill the flattened data center topology.

2.3.5 BCube and PCube

BCube [56] is a server centric data center network architecture which relies on a recursively defined structure consist of two types of devices: multi-port servers, typically no more than four ports, and multiple layer of commodity switches. In BCube, servers are not only hosts but also relay nodes for each other and take part in traffic forwarding through multiple parallel short paths. The design is driven by demands for intensive computing and higher bandwidth requirements to support applications for different traffic patterns. BCube supports and accelerates all types of traffic patterns and provides high network capacity due to its low diameter. The benefits of BCube design includes providing fault tolerance and load balancing, while requiring lower cooling and manufacturing cost.

PCube [65] is an elastic data center structure designed to optimize power efficiency through scaling energy consumption according to traffic demand patterns. PCube

powers on/off some switches dynamically according to the network bandwidth demand. Adjustment is accomplished in a way that there are always multiple paths between any two pair of servers to obtain fault tolerance. It can be directly applied to existing hypercube structured data center networks, e.g. BCube, without any hardware modification or rewiring.

2.3.6 DCell

This is a scalable server centric data center network topology architecture providing rich connectivity among servers via commodity switches [57], but additional wiring is required to connect the servers and switches, specially in long paths. DCell is scalable due to its recursive structure, which allows extending the network gradually without extra rewiring or address changing.

2.3.7 CamCube

CamCube is a pure server-centric data center network architecture designed for container data centers [41] based on torus topology to connect a single server with six other servers using network interfaces. Network traffic routing is performed by servers while network switches are not used at all in the CamCube data center network architecture. The CamCube uses different routing schemes for different services and provides a load balanced flow assignment. It eliminates the use of similar routing logic for all services, and facilitates the use of customized routing logic for individual services.

2.3.8 Optical data center network

Since optical network devices consume less power with orders of magnitude compared to electric networks, some efforts has been made to move the data center networks toward optical technology, which provides high speed, bandwidth and less complexity. Some topologies such as C-through [134] and Helios [47] are designed

to upgrade the current data center network topology based on commercially ready optical network devices, combined with the electric switch centric architecture.

In a nutshell, optical data centers can be more energy efficient compared to the other topologies. Moreover, flattening the data center topology can reduce the energy consumption thanks to fewer number of switches as well as reduced network diameter and average path length [4, 81].

2.4 Virtualization

Here, we address virtualization as a driving force of cloud computing, first in the context of available techniques, and then integrated with the peer-to-peer deployments.

2.4.1 Virtualization Technologies

With the advent of server-side computing as a service, provisioning resource guarantees and isolation in multi-tenant environment became of utmost importance. It became imperative that these infrastructures satisfy the goal of application isolation and resource efficiency. In order to achieve these goals, the infrastructure economics must allow servers to be shared among multiple users and at the same time guarantee operational isolation of applications. Virtualization is the most widely adopted solution to guarantee these goals. Consolidation by using virtualization leads to application isolation, better resource utilization and lower operational costs. Different virtualization technologies are summarized in Table 2.1. According to [103] there are two types of virtualization: Type1 and Type2. However, to be able to classify the virtualization techniques without any overlap in their characteristics we divide Type1 into two different classes. Moreover, classic classification does not include the recent technology, i.e. container virtualization, which we address in this work.

TABLE 2.1: Virtualization Technologies

1whiteblack **Type (Classic Definition)**	**Layer**	**Virtual Machine Monitor**	**Virtualization Technology**	**Example**
Type 1	VMM	ISA(HW)	native virtualization	XenServer
Type 1-modified	ISA(HW) and hypercalls	hosted	para-virtualization	Hyper-V, VMware ESX(i)
Type 2	ISA(HW)	hosted	full virtualization	VMware Workstation, virtual box
Container Virtualization	ABI(OS)	kernel	container virtualization	LXC, Docker

Para-virtualization (Type 1- modified) is a sort of virtualization in which the guest operating system (the one being virtualized) is aware that it is a guest and accordingly has drivers that, instead of issuing hardware commands, issue commands to the virtual machine monitor. This includes memory and thread management as well, which usually require unavailable privileged instructions in the processor. On the other hand, in full virtualization (Type 2), the guest operating system is unaware that it is in a virtualized environment, and therefore hardware is virtualized by the host operating system so that the guest can issue commands to what it thinks is actual hardware, but really are just simulated hardware devices created by the host.

Native virtualization (Type 1) is a type of full Virtualization where the microprocessor architecture has special instructions to aid the virtualization of hardware. These instructions might allow a virtual context to be setup so that the guest can execute privileged instructions directly on the processor without affecting the host. Such a feature set is often called a virtual machine monitor. If said instructions do not exist, full virtualization is still possible; however, it must be done via software techniques such as dynamic recompilation where the host re-compiles on the fly privileged instructions in the guest to be able to run in a non-privileged way on the host.

Mostly, cloud based solutions for virtualization rely on Hypervisor based virtualization (Type 1), such as Xen [11], since it can support different flavours of operating systems such as Linux, Windows, etc. Additionally, with the proliferation of hardware virtualization in most modern architectures, this allows guest operating systems to run unmodified. However, Hypervisor based virtualization can suffer from performance degradation as they incur an additional overhead of VM management by the Hypervisor, in Instruction Set Architecture (ISA).

Therefore, hosted virtualization (Type 2) is also becoming a widely adopted alternative since each guest can have its own kernel and the host contains a modified kernel with extensions to manage and run multiple VMs.

Container virtualization, on the other hand, can execute applications at near native speed since they have no additional layer of routing and share the same kernel. This type of virtualization, virtualizes the Application Binary Interface (ABI) layer. Nonetheless, since the kernel is shared, they can only run guest operating systems that support the host kernel.

LXC (Linux Container) [64] is a common instance of container virtualization. LXC is an operating-system-level virtualization method for running multiple isolated Linux systems (containers) on a control host using a single Linux kernel. Docker can also use LXC as one of its execution drivers, enabling image management and providing deployment services. Docker containers wrap up a piece of software in a complete filesystem that contains everything it needs to run: code, runtime, system tools, system libraries, i.e. anything you can install on a server. This guarantees that it will always run the same, regardless of the native environment it is running in.

Virtualization density refers to the number of virtual machines a physical host can maintain, while providing enough compute resources for every virtual machine to perform well. It depends on multiple factors such as: server hardware, virtualization software, service type and workload diversity. These varying factors make it difficult to come up with an absolute number for virtual machine density across all scenarios. Driving up the VM density reduces the cost incurred but at the same time introduces additional challenges in guaranteeing performance because of contention in shared resources. Typically the bottleneck manifests in the memory subsystem and the amount of available memory. Commodity machines are scarce is such resources and as such the system will benefit from conservative provisioning in order to provide best-effort guarantees.

2.4.2 Virtualization in P2P context

Although multiple virtualization techniques exist, virtualization in the context of P2P presents specific challenges that need to be addressed. P2P is comprised of commodity machines that are not server grade and as a result do not have high computational capabilities. In data centers, typical virtualization technology is hypervisor based; hosted virtualization is becoming prevalent, too. However, P2P servers are limited by the available resources, and it becomes imperative to ensure that virtualization does not impose a high overhead on these resource limited machines.

Hypervisor based virtualization techniques like Xen impose a high overhead because of an additional level of routing at the expense of running any operating system. Container virtualization is, thus, a better fit for such environments with limited resources, due to its near native speed with little overhead. Hosted virtualization is also an attractive alternative for resource limited environments, since it imposes minimal overhead. However, each guest can have its own kernel and this has a performance impact when spawning a new VM. LXC has lesser overhead from this perspective. Despite the host of relative advantages and disadvantages, LXC appears to be a reasonable choice of virtualization technique to adopt in a P2P environment.

2.5 Smart Grid

A smart grid is an electrical grid which includes a variety of operational and energy measurements including smart meters, smart appliances, and leveraging renewable energy resources [97].

Since the early 21st century, opportunities to take advantage of improvements in electronic communication technology to address the limitations and costs of the electrical grid have emerged. Technological limitations on metering no longer force peak power prices to be averaged out and passed on to all consumers equally.

In parallel, growing concerns over environmental damage from fossil-fired power stations has led to a desire to use large amounts of renewable energy. Therefore, a more sophisticated control systems to facilitate the connection of sources to a highly controllable grid is required [46].

Micro grids are modern, localized, small-scale grids, contrary to the traditional, centralized electricity grid (macro grid). Micro grids can disconnect from the centralized grid and operate autonomously, promote grid resilience and help mitigate grid disturbances. They are typically installed by the community they serve. Micro grids exert diverse distributed energy resources, such as solar hybrid power systems, which reduce the amount of emitted carbon significantly.

Smart grid enables the industry's best ideas for grid modernization to achieve their full potential and prepares a cleaner and more efficient, reliable, resilient and responsive electric system. A smart grid system requires a highly responsive monitoring capability suitable for wide area deployments. It needs a large scale infrastructure for collecting and communicating data; likewise, it must have access to flexible (possibly network-scattered) computational power, network bandwidth, and storage capacity. The distributed nature of data sources, the possibility that data may need to be collected from multiple (competitive) power producing and transport enterprises, and the need for timely state estimation, all make the system more complicated. An out-of-the-box infrastructure solution to address all the smart grid computational infrastructure is the state-of-the-art Information and Communication Technology (ICT) infrastructure.

2.6 Summary

In this chapter, we reviewed the background concept that we form the reset of the thesis on. First, diverse range of cloud platforms with different granularity are outlined. Then, data center networking topologies are surveyed. The state-of-the-art virtualization technologies, as the cornerstone of cloud computing paradigm,

as well as their suitability for P2P-cloud are speculated. We closed this chapter by defining smart grid concept, its requirements and how ICT can serve these needs.

Part II

Analysing Energy Efficiency of P2P-assisted Cloud

—"All models are wrong, but some are useful."

-George Box

3

Energy Analysis Metric

The focus on energy management has been a cross-cutting concern across various computing disciplines including computer architecture (hardware design), hypervisors, operating systems and system software [18]. Figure 3.1 captures the various techniques developed to reduce energy consumption across the service provisioning stack. In each layer of this stack, power and performance may be mapped to different metrics, which formulate power and performance with distinct granularity.

By accessing the limited set of data in different layers of the stack, we can still analyze the energy consumption with different levels of accuracy. In this chapter, we sketch a framework to formulate energy consumption from different perspectives all through the service provisioning stack. We address the challenges of energy modeling in each layer with associated granularity. In other words, the granularity in the top levels is coarser, due to access to the coarse metrics. However,

the further we move toward the lower layers in the service stack, the finer the estimation granularity is. Moreover, we introduce the energy complexity metric to model the energy consumption in application layer which is hardware and lower layers agnostic. Generally, the energy model offered in each layer is lower layer(s) agnostic, but it is aware of the upper layers' properties.

What is more, the growing body of work on power analysis only aims at reducing the total power consumption in the infrastructure, without taking into account the energy-related behavior of each individual job, its performance and price, i.e., how expensive and efficient is the energy employed for the observed job performance or progress. State-of-the-art work emphasizes on modeling of power dissipation, and decides on the energy efficiency from a power efficient system view. However, power efficient systems may not always save energy for running the same workload, due to the plunge in the system performance. For instance, if the performance is translated to the execution time, and the power reduction is reached by slowing down the process time, the applications may reside longer in the system and this may result in the more overall energy dissipation, since energy is the cumulative power dissipated over the execution time.

In the procedure of decreasing the energy consumption of cloud services, we may end up with this pitfall that the energy consumption decreases remarkably, but the performance is violated at the same time. Users may leave a system if they do not get the desired quality of service (QoS). Indeed, we need to develop a more comprehensive framework to provision QoS for a diverse range of services and applications using collaborative environments. In this chapter, we introduce the *Energy Effectiveness* metric [113] to include performance in our energy analysis, as well. Then, we sketch energy and power model in each layer of the service provisioning stack, from hardware to virtualization and application layers.

This chapter is structured in a top-down format as the following. In Section 3.1, we outline the background and related work that we build our contributions of the chapter on. We introduce Energy Effectiveness as a performance aware energy analysis metric in Section 3.2. Then, we address the energy and power modeling

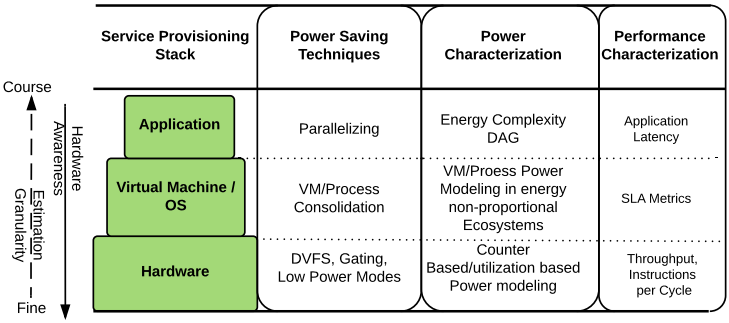


FIGURE 3.1: Energy and performance conceptualization across the service provisioning stack

in hardware, virtualization and application layers, as the core requirement to formulate energy effectiveness, in Sections 3.3, 3.4, and 3.5 respectively. We explain the evaluation framework and experiment setup in Section 3.6. Section 3.7 verifies all the arguments and proposals in the chapter via an evaluation framework. The chapter is closed in Section 3.8.

3.1 Background and Related Work

In this section, we introduce the service provisioning stack. Moreover, the overview of previous research on power modeling of all layers of the stack is given in this section.

3.1.1 Service Provisioning Stack

Three main components involved in the service provisioning are the application, operating system and hardware. In virtualized platforms, operating system is replaced by the Virtual Machine (VM), as illustrated in Figure 3.1. To execute a service on the cloud platform, we need to run the associated application(s) on the VM which is virtualizing the hardware resources of the hardware platform which it is running on. In each layer of this stack, power and performance may be defined as

different metrics. Namely, in application layer, performance is generally translated to latency, whereas in VMs it is mapped to SLA metrics and throughput is the interpretation of performance in the I/O hardware level. Also instructions per cycle indicate the performance in the CPU hardware, as mentioned in Figure 3.1.

All the same, across the stack, we need a translation to hardware agnostic power model in the application layer, and partly in VM/OS level, which should be mapped to a hardware aware model at the bottom of the stack at runtime. In the rest of this section, we review the state-of-the-art power characterization techniques across the service provisioning stack.

3.1.2 Energy Proportionality

The vision of an energy proportional system implies the power model of an ideal system in which no power is used by idle systems ($P_s = 0$), and dynamic power dissipation is linearly proportional to the system load [12].

LDR indicates the difference of the actual power consumption, $P(U)$, and linear power model over the linear power model as in (3.1). P_d represents dynamic power dissipation, which is proportional to the utilization level.

$$LDR = \frac{P(U) - (P_s + P_d)}{P_s + P_d} \tag{3.1}$$

IPR is the indicator of idle to peak power consumption (P_{idle}/P_{Max}) as illustrated in (3.2).

$$IPR = \frac{P_{idle}}{P_{Max}} \tag{3.2}$$

To measure how far a system power model is from the ideal (energy proportional) one, Proportionality Gap(PG) [137] is defined as the normalized difference of the real power value and the ideal power value, which is indicated as $P_{Max} \times U$, under a certain utilization level as shown in (3.3). Therefore, having proportionality gap

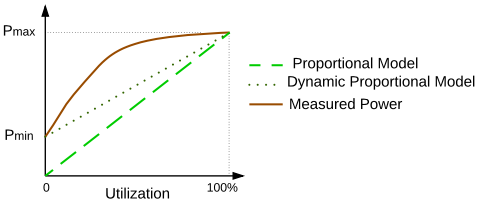


FIGURE 3.2: Energy Proportionality

values for a given device, we can reconstruct the power model of the device.

$$PG(U) = \frac{P(U) - (P_{Max} \times U)}{P_{Max}} \tag{3.3}$$

Given the state-of-the-art hardware, designing hardware which is fully energy proportional remains an open challenge; the power model of an energy non-proportional system is illustrated in Figure 3.2. However, even in the absence of redesigned hardware, we can approximate the behavior of energy proportional systems by leveraging combined power saving mechanisms [128] and engaging heterogeneous commodity devices combined with powerful server machines in lieu of the homogeneous server hardware platform [137]. We address the effect of heterogeneous hardware on energy proportionality in the next chapter.

3.1.3 Power Characterization

Research in power modeling can be broadly classified into (i) Simulator-based, (ii) CPU Utilization-based (iii) Event or performance counters based and (iv) Coarse-grained. Early power management research exerted analytic power models based on voltage and frequency [15], which are fast, but only provide rough estimates. Coarse-grained estimation based on the type and state (active, off) of the processor have been used in [19]. However, with the increase in the dynamic power range of servers, a more accurate power prediction method is required.

3.1.4 Hardware Power Modeling

As current platforms do not provide fine-grained power measurement capabilities, power models are the first step to enable the dynamic power management to reach the power proportionality on all levels of a system.

A CPU utilization based model is one of the most popular power estimation models used in practice [52]. However, with the increase in the dynamic power range of servers, a more accurate power prediction method is needed. Interestingly, the workload-sensitive nature of CPU-based models has been recently cited as a reason to go back to using detailed event counters for predicting processor and memory power usage under voltage scaling [23].

Currently, the approach closest to the hardware-based monitoring is the running average power limit (RAPL) feature available for the Intel Nehalem, Sandy Bridge, Ivy Bridge and Skylake CPUs [59], which allows to monitor the power consumption of the whole CPU package. As this feature is not available on the other CPUs such as AMD, generic power models typically rely on a number of performance counters, allied with regression techniques to define a set of counters as the representative of different resources.

Counter based power models follow either top-down or bottom-up approaches [23]. The top-down approach aims to define simple, fast and easy to deploy models which are architecture agnostic. This approach models the power using a reduced set of performance counters and avoids the requirement of specific knowledge of the architecture.

The bottom-up approaches rely on some knowledge of the underlying architecture to produce more informative, responsive and accurate power models. These approaches are able to decompose the power components of the architecture but they are more complex to deploy and less portable compared to the top-down ones.

Usually the accuracy of the models is validated by comparing estimates with the measures of a power meter when running benchmarks in isolation [135].

Power modeling often considers learning techniques such as sampling [21] that assume the proportionality of the system events to the power consumption. Measurements of a hardware power meter are gathered and subsequently used, together with a set of normalized estimated values, in various regression models, which are so far mostly linear [89]. However, in [131], it is stated that linear power models, by depending on the CPU load are not sufficient anymore and more parameters have to be considered. The study in [89] shows that, especially in multi-core systems, linear models lead to a much higher mean relative error for CPU power consumption and cannot easily be improved by applying more complex techniques. Linear models rely on the independence of the covered features, which is not realistic in current systems. Polynomial/exponential regression can cover these dependencies and, as shown in [26], a quadratic solution better fits the power modeling of multi-core systems.

The described systems must, however, isolate processor features, such as hyperthreading and turbo-boost, to avoid hidden states. HAPPY [141] introduces a hyperthread-aware power model that differentiates between the cases where either single or both hardware threads of a core are in use. In [22], a bottom-up approach is introduced that covers simultaneous multi-thread and multi-core effects in the power modeling.

As a recent work in this line, BitWatts [39] introduced a counter based power model for each individual frequency, but it does not consider the multi-core and hyperthread effects. VPM [107] as another recent work, proposes a power model, which supports multicore and heterogeneous architectures. In a nutshell, a reasonable server power model should cover the modern CPU architectures, which follow multi-core and heterogeneous cores of the CPU, and are equipped with dynamic overclocking (turboboost), dynamic scaling, and hyperthreading. Moreover, it should be architecture-agnostic and hardware oblivious.

3.1.5 VM Power Modeling

In multi-tenant ecosystems, the efficiency of VM consolidation, power dependent cost modeling, and power provisioning are highly dependent on accurate power models. Such models are particularly needed because it is not possible to attach a power meter to a virtual machine.

VMs can be monitored as black-box systems for the coarse-grained metering and scheduling decisions. However, for the fine-grained metering and scheduling decisions, specially with the heterogeneous hardware, finer-grained estimation at the sub-system level is required. The fine-grained power estimation of the VMs requires profiling each application separately. To exemplify, WattApp [77] relies on the application throughput instead of the performance counters as the basis for the power model.

To generate a VM power estimator, JouleMeter [71] assumes that each VM only hosts a single application, thus, treats VMs as black boxes. In a multi-VM system, they try to compute the resource usage of each VM in isolation and feed the resulting values in a power model. VMETER [28] estimates the consumption of all active VMs on a system. A linear model is applied to compute the VMs' power consumption exploiting the processor utilization and I/O accesses from each physical node. The total power consumption is subsequently computed by summing the VMs' consumption with the power consumed by the infrastructure. However, this method fails to capture the effect of the energy non-proportional hardware.

Bertran et al. [23] proposed a performance-monitoring counters (PMCs) approach to employ a sampling phase to gather data related to compute energy models from these samples. With the gathered power models, it is possible to predict the power consumption of a process, therefore, apply it to estimate the power consumption of the entire VM. All the same, in [76] a counter based VM power modeling solution is proposed which considers each VM as a process running on the host machine. In [131], a holistic approach toward the VM resource consumption is considered,

which maps different VM resource utilizations to the energy consumption applying polynomial regression techniques. This model, however, overestimates the VM power consumption.

As a recent work in this line, BITWATTS [39] is a middleware solution to estimate the power consumption of the software processes running in virtualized environments. It provides a process level power model, which is application agnostic and accounts for virtualization by exposing power probes from the host operating system (OS) to the guest OS. Therefore, BITWATTS can estimate the power consumption of processes running within a VM. In addition, this design can operate in distributed settings, with consumption information transmitted over high-performance publish/subscribe middleware.

To sum up, there is limited work centered on VM power and energy modeling. The state-of-the-art models do not take into account the effect of energy non-proportionality or performance interference. In this work, we circumvent these limitations and propose a power model that is aware of both energy non-proportionality and performance interference.

3.1.6 Application Power Characterization

To the best of our knowledge, so far, there is little effort on application energy characterization. A line of work is toward profiling applications to figure out the energy consumption pattern of a particular application. In [112], a counter based application resource usage profiling is proposed, which is followed by a mechanism to map it to the energy consumption. In [35], a fine grained application energy profiling is proposed to enable application developers to make energy efficient choices.

Another research direction is toward computing relative power values for the applications running on a host. This means that the power is not estimated in watts, but it is calculated as a function of the resource utilization in the system.

Namely, Mac Activity monitor [1], reports the power impact for each application as a function of the CPU utilization and the idle state time of an application. Green Tracker [7] follows the same approach to give an insight on the energy consumption of each software running.

Tangential to this goal, another line of work is attempting to profile the application energy consumption for mobile devices [8, 62]. They try to characterize the diverse resources of mobile devices such as GPS, WiFi, CPU, memory and storage requirements of individual mobile applications. All the proposed techniques are measurement based, which are completely hardware dependent. However, we need a hardware oblivious model to fulfill the hardware agnosticism property in the application layer of the stack.

In [109] a new complexity model is introduced to account for the energy used by an algorithm. Based on an abstract memory model (which was inspired by the popular DDR3 memory model), they present a simple energy model that is a (weighted) sum of the time complexity of the algorithm and the number of 'parallel' I/O accesses made by the algorithm. In their next work [110], they experimentally validate the algorithm energy complexity model derived. This energy complexity model is asymptotic which is expected in a hardware agnostic conceptualization. However, this work only covers algorithm level energy formulation. Whereas, application energy modeling is more complicated, because an application is constituted of algorithms and data flows.

3.2 Performance-aware Energy Analysis Metric

As aforementioned, if the service is not delivered as expected, it may tarnish the provider's reputation. Thus, it is required to obtain a service with desirable response time as well as acceptable throughput, availability and consistency level. Attaining high QoS may impose more energy consumption. Therefore, we should strive to alleviate the burden of high service energy. To this end, the energy

[1] `https://support.apple.com/en-us/HT201464`

efficiency is introduced in [125] as $\frac{Performance}{Energy}$. However, in this metric, there is no mechanism to guarantee the performance, and all sensitive and non-sensitive services are treated equally. For instance, if we over provision the performance, for a particular service, we probably have to spend more energy, while gaining nothing in exchange, since it is not perceived by the user. However, energy efficiency ratio may increase in this scenario.

Increasingly, the most efficient servers nowadays, consume at least 20-30 percent of their nominal power in the idle case, and deviate from linear proportionality property noticeably according to the SPECPower_ssj2008 [2]. Hence, Idle to Peak Ratio (IPR) and Linear Deviation Ratio (LDR), for the current power models, are still remarkably higher than the ideal case. Higher IPR encourages the server consolidation for the sake of power saving; however, this is not always a solution. Utilizing a server to its 100% capacity may affect the application performance tremendously, thus reducing actual energy efficiency of jobs, and also does not contribute to power saving in cases that LDR is unequal to one and when the interference overhead exceeds the proportion of static power.

Moreover, co-location of applications has its own challenges. Workload intensity is often highly dynamic. The power profile of the data center hardware is inherently heterogeneous; this makes the optimal performance gain problem more complicated. The non-linearity, and in some cases unpredictability, of the energy efficiency profile currently aggravates the complexity of energy efficient co-location management. Therefore, to address the performance awareness in the energy analysis, we devise *energy effectiveness* metric.

3.2.1 Energy Effectiveness

The concept of **energy effectiveness** introduces a metric to achieve conciliating two goals of performance guarantee and energy saving. Thus, Energy Effectiveness, as proposed in (3.4), is a speculative metric that quantifies the degree to which

[2] `https://www.spec.org/power_ssj2008/`

the ecosystem is successful in decreasing energy dissipated while the performance is not significantly violated.

$$\mathfrak{E} = \alpha \times \frac{E^*}{\hat{E}} + (1 - \alpha).min(1, \frac{\hat{\mathcal{P}}}{\mathcal{P}^*}) \tag{3.4}$$

In (3.4), $\mathfrak{E}$ introduces the energy effectiveness, $\hat{E}$ and $\hat{\mathcal{P}}$ stand for the estimated or measured energy consumption and performance of the considered service on the running platform or in a standalone analysis, depending on the context (i.e. layer of the stack). Note that in the application layer, $\hat{E}$ is translated to *Energy Complexity* ϵ, which characterizes the energy consumption pattern of the application. More detailed description of energy complexity is available in Section 3.5.

E^* factorizes to the energy consumed in order to provide the service in an energy proportional system with a linear power model, representing the equality of utilization and associated power dissipation ($P(U) = U$), this is the minimum reachable energy consumption. Again in application layer, this is equivalent to ε associated to energy complexity that is introduced in Section 3.5. $\mathcal{P}^*$ is quantified based on the Service Level Objectives(SLO) and Service Level Agreement(SLA) parameters depending on the interpretation of the performance on each layer of the service stack. Quantifying the SLA metrics is extensively studied in [50]. In other words, $\mathcal{P}^*$ represents the desirable performance conceptualized in the associated service stack layer.

Moreover, it is necessary to handle the trade off among these tightly coupled parameters to achieve an efficient mechanism. Intuitively, an adaptive model, covering the system and user requirements, is appropriate for this purpose, because the parameters are tunable in such a model. The model supports a more diverse range of cases due to its flexibility. Therefore, we introduce α as the adaptiveness parameter. Based on the performance sensitivity of the applications, we can tune the α in the range of 0 to 1 to weight the energy and performance accordingly.

3.2.2 Vulnerability Factor

Further to energy effectiveness, we define **Vulnerability Factor**, $\mathcal{V}$, which embodies to the range of variability in the energy effectiveness as $\mathcal{V} = \frac{\partial \mathfrak{E}}{\partial \alpha}$. Namely, $\mathcal{V}$ represents the slope of the $\mathfrak{E}$ equation when α varies in range of 0-1. The higher the $\mathcal{V}$, the more influence the adaptiveness factor has in the $\mathfrak{E}$ value, and the more important is to set it properly. $\mathcal{V}$ can be determined in the SLAs according to the user incentives (previously addressed for cycle-sharing [108]) and service requirements (previously addressed for virtual machines [122] and Java applications [121]).

The energy effectiveness metric we define here has a bounded value in the range of 0 to 1 for the sequential processing and interactive applications such as live streaming, while this value can exceed one in case of parallel processing applications, e.g. MapReduce. This value tightly couples with the level of parallelism and the energy proportionality of the host platform. Quantifying the correlation of the parallelism and energy effectiveness is beyond the scope of this work, but interested readers may refer to [109] to find out more.

3.2.3 Relative Energy Effectiveness

While energy effectiveness defines how much the ecosystem can be improved to reach the energy proportional system, where guaranteeing the performance, a relative metric is also needed to compare the energy conservativeness and performance guarantee among diverse range of systems.

To fulfill this requirement, we define $\mathfrak{E}^R$ as the **Relative Energy Effectiveness** which characterizes the energy effectiveness metric for the sake of comparison with other hardware platforms, VMs or applications in a standalone manner. $\mathfrak{E}^{\mathbb{R}}$ is formulated as (3.5). Compared to $\mathfrak{E}$ equation, in relative energy effectiveness, adaptiveness factor α is omitted for simplicity. Moreover, in this definition, we do not normalize estimated energy $\hat{E}$ by energy proportional value, since this value may vary for two comparing platforms.

$$\mathfrak{E}^{\mathbb{R}} = \frac{1}{\hat{E}} + min(1, \frac{\hat{\mathcal{P}}}{\mathcal{P}^*}) \tag{3.5}$$

3.3 Hardware Power Model

The power drawn in a hardware element P_{hw} is a combination of the static power P_s and dynamic power P_d. P_{Max} indicates nominal power as the maximum power device can dissipate at utilization level U. Static power is consumed even if the machine is idle, while the dynamic power is proportional to the resource utilization within the host.

$$P_{hw} = P_s + (P_{Max} - P_s) \times U \tag{3.6}$$

In (3.6) a linear power model is outlined, where a linear correlation among the utilization level and the power drain is assumed. However, in real systems, the LDR is not equal to one. Therefore, a more complicated model is required to formulate power dissipation as discussed in Section 3.1. Having access to a power model, we can formulate energy dissipated via dividing the power by throughput τ, since elapsed time is reversely proportional to the throughput ($t = \frac{1}{\tau}$).

Power is majorly drawn in the communication and processing hardware, during the service provisioning life-cycle. However, for data centers this list extends to the cooling, lighting and maintenance energy as defined in the data center Power Usage Efficiency (PUE) [14]. To obviate the power efficiency of a data center, PUE parameter is defined as the ratio of total amount of power used by a computer data center facility to the power delivered to the computing equipment. The higher the PUE, the less power efficient the data center is. The energy overhead of all the non-IT devices such as cooling, lighting, etc. is typically modeled as $\frac{(PUE-1)}{U}$ coefficient of the overall power of the resource [9, 87], where U represents the utilization of the resource. In this section, we model power and energy dissipation in the host and communication hardware elements. Note that host power modeling is not a major contribution of this thesis; however, for the sake of completeness,

we sketch a general counter based power model in this chapter. The more accurate the hardware power model, the more accurate the power estimation for VM and application. Nonetheless, the power model needs to be architecture agnostic to keep the generality of the framework.

3.3.1 Host Power Model

Although the power drawn in the host is measurable through attaching a physical power meter to the system, in order to be able to map, correctly and accurately, the power dissipated in the VMs and applications to the corresponding host power consumption, we need to model the server power dissipation pattern.

To formulate a fine-grained power model accurately, relying on performance counters is the best of the state-of-the-art. The methodology proposed in [23] is the one we develop our model based on. Note that to keep the generality of the framework we follow top-down approach in power modeling and only consider the generic counters. The model following this approach is not the most accurate model, but we need to make a trade off between accuracy and generality. Typically the following steps should be taken to make a counter based power model.

1. A specific set of the micro-benchmarks which are designed to utilize each resource element (i.e. CPU, memory and I/O) in different levels should be executed on the machine. We designed a set of micro-benchmarks that is discussed in Section 3.6.

2. Generic performance counters as well as utilization data should be collected as long as the benchmarks are running on the machine. We list the studied performance counters in Section 3.6.

Therefore, the energy dissipated in the host can be calculated using the following formula.

$$\hat{E} = \frac{PUE \times P_{host}}{\tau} \tag{3.7}$$

3.3.2 Communication Power Model

Communication power P_c, in general for a bit of data, can be formulated as $P_c = \sum_{h=1}^{hops}(P_{switch}(h) + P_{host}(h))$. Where *hops* represents the number of hops that should be traversed between source and destination. $P_{switch}(h)$ and $P_{host}(h)$ represent the power drawn in the switch/router that forwards/routes the data to the next hop and the power dissipated in the host for the same hop in case of server centric switching. In the switch centric communication within a data center, switches that connect the hosts are the major power consumption sources. In the pure server centric data center networks, servers are in charge of forwarding the data; thus, communication energy is added to the server energy profile further to the processing energy. For the hybrid network topologies, communication energy is partly dissipated in the switch and partly in the servers.

In (3.8), multiplying the PUE, data size S_{data}, and replication factors r divided by throughput τ_c [3], communication energy is derived from the communication power model. $\varphi(i)$ represents the oversubscription ratio in hop i.

$$E_c = (PUE \times r \times S_{data}) \sum_{i=1}^{h} \frac{P_{switch}(i) + P_{host}(i)}{\tau_c(i)\varphi(i)} \tag{3.8}$$

Moreover, the network topology impacts the power usage profile. Therefore, in the rest of this section, we discuss communication power modeling in the intra-data center communication, P2P communication and Internet communication.

Data center communication power: Here, we study the power consumption of a three-tier, hierarchical topology. The motivation behind formulating the hierarchical model is that it can be easily generalized to numerous intra-data center topologies, e.g. Fat-Tree[5] , VL2[53], BCube[56], PCube[65], etc. The tree depth is defined based on the path messages should traverse within the data center in each layer. For the topologies which deviate from this property, e.g. CamCube[41],

[3]Note that throughput here is different than bandwidth which refers to nominal network capacity. Throughput considers delay and overhead, besides to the nominal capacity.

we analyze the energy model separately. We assume an l level tree in which hosts are in the leaves and are connected to an edge switch as their predecessor via Gigabit Ethernet links. The edge switches are connected via an aggregate switch; this process proceeds in two or more levels to create the root of the tree.

To assign a task to a host, the root aggregate switch transmits the task data to the selected host through the tree. Assuming the homogeneous switches in each level of the tree, the power consumed for this purpose is calculated as in (3.9). P_{switch} stands for the power drawn by the switch. Additionally, we added P_{host} to each level's consumption to generalize our model.

$$P_c^{intra-DC} = \sum_{i=1}^{l-1} (P_{switch}(i) + P_{host}(i)) \tag{3.9}$$

Therefore, in a switch centric model, $P_{host} = 0$, while in a pure server centric model $P_{switch} = 0$ and in a hybrid model, power is drawn both in switches and servers.

Referring to (3.9), the depth of the tree, l, directly influences the power efficiency of the data center. The tree depth is determined by the number of hosts and network topology. The larger the data center is, the bigger the number of the switches and links required to connect the hosts and the deeper the tree is. Furthermore, flatter data center topologies, such as flattened butterfly [4] and FlatNet [81], obtain shorter paths via less switches. Topologies providing smaller network diameter are also more energy efficient due to shorter average path that should be traversed among the servers.

Therefore, smaller distributed data centers, serving the users independently, are more power efficient than a single mega-data center model, following a tree intra-data center topology. Loosely paraphrasing, in small data centers, the network diameter is smaller, since the number of switches and links required to connect the hosts within a data center is directly related to the number of hosts.

P2P-cloud communication power modeling: As described in the background chapter, we assume a P2P-cloud deployed in a community network. Inexpensive Wi-Fi devices have made the deployment of such communities feasible in recent years. Some flourishing instances are Guifi.net [4], with more than 20,000 active nodes, Athens Wireless Metropolitan Network [5], FunkFeuer [6], Freifunk [7], etc.

In these networks, hosts within a vicinity are usually connected via wireless links that form a wireless network. Thus, the power consumed for communication within a vicinity predominantly embraces the wireless network power consumed to transmit data [48].

Community networks are rather diverse in terms of size, topology and organization. This is a consequence of their unplanned deployment, based on the cooperation of their own customers/users; therefore, characterizing the power consumption in these networks is challenging. However, in the big picture, the energy consumption in P2P-communication platform manifests from the number of hops to be traversed to reach a particular peer, energy dissipated in each intermediate hop infrastructure such as switches, routers and antennas, and the data size S_{data} and replication factor r as shown in (4.3). Dividing these terms by network throughput τ_h, we can calculate the energy dissipated for data transfer, since elapsed time is inversely proportional to throughput ($t = \frac{1}{\tau_{P2P}}$). The power consumption is characterized in the P2P-cloud through measurement in a production wireless community network later in this chapter, in Section 3.6.

$$E_c^{P2P} = \sum_{h \in hops} P_c(h) \times \frac{r \times S_{data}}{\tau_h} \tag{3.10}$$

Internet power consumption: P2P-clouds for inter-vicinity communication and classic data centers for communication with users rely on the Internet. Thus, to analyze the energy consumption of these systems, we should be aware of the

[4] `http://guifi.net/en`
[5] `http://www.awmn.net`
[6] `http://www.funkfeuer.at`
[7] `http://freifunk.net`

Internet energy consumption as well. Power drawn in the Internet is subject to the hardware and distances exploited. The Internet infrastructures are classified as core, distribution and access. Core layer includes the Internet backbone infrastructures such as fiber-optic channels, high speed switch/routers, etc. Distribution infrastructures play role as intermediaries to connect the Internet Service Providers (ISPs) to the core network. The access layer connects the user to ISP communication infrastructure.

Since there is a diverse range of hardware in each layer, it is not trivial to form a comprehensive analysis on energy consumption of the Internet. However, Baliga, et al. [10] conducted a study on the prevalent Internet hardware energy consumption. We rely on this study for the Internet power consumption part of our analysis by driving the model in (3.11). In this model, $P_{Internet}$ stands for the Internet power consumption which is a combination of power drawn in each level $L = \{core, distribution, access\}$. $P_c(l)$ denotes router power consumption in layer l, and $|hops(l)|$ indicates the number of hops, as the cardinality of the hops set in layer l, should be traversed at l layer.

$$P_{Internet} = \frac{1}{\varphi} \times \sum_{l \in L} P_c(l) \times |hops(l)| \tag{3.11}$$

In (3.12) energy consumption of communication over the Internet is modeled by dividing the power in each Internet layer by the throughput at that level τ_l.

$$E_{Internet} = \frac{1}{\varphi} \times \sum_{l \in L} \frac{P_c(l) \times |hops(l)|}{\tau_l} \tag{3.12}$$

The concept of oversubscription, φ, exists in the Internet communication, where Internet service providers exert it as a strategy to utilize the resources by overbooking the shared infrastructure among the users. The more the resources are shared temporally, the less the energy consumption is due to the shared static power dissipated. Oversubscription for the home users is 40:1 and for the business connection is around 20:1 in the current Internet.

3.4 VM Power Estimation

As aforementioned, direct VM power measurement is not possible, therefore, VM power modeling is essential to estimate VM power consumption. Models for power estimation have been majorly studied at the level of processors, and less extensively in the context of virtualization.

Co-location of VMs faces several challenges. Workload intensity is often highly dynamic. The power profile of the data center hardware is inherently heterogeneous; this makes the optimal performance gain problem more challenging. The non-linearity and in some cases unpredictability of the energy efficiency profile aggravates the complexity of the energy efficient VM consolidation. In this section, we elaborate the challenges in the VM power modeling.

3.4.1 Interference Overhead Modeling

Energy consumption of the host per VM comprises of both static power and dynamic power consumed by the VM. While static power is independent of resource utilization, dynamic power is proportional to the resource utilization. However, in a multi-tenant setting, dynamic power of a VM is also influenced by the overhead caused in the hypervisor and performance interference from other VMs due to the contention in shared resources. Estimating this overhead is complicated since the pattern of the hypervisor overhead and performance interference is tightly coupled with the number of VMs, the type of resources each VM asks for, and the number of times the switching occurs between VMs. Thus, for a more accurate estimation, a VM power model should also take into account hypervisor overhead and performance interference.

In [124], the authors argue that, in the virtualized environments, energy monitoring has to be integrated within the VM as well as the hypervisor. They assume that each device driver is able to expose the power consumption of the corresponding device as well as an energy-aware guest operating system and is limited to the

integer applications. Work in [104] introduces an interference coefficient, defined to model the energy interference as a separate implicit task. Interference energy is estimated as the coefficient of the summation of the idle and isolated run for each VM. The major contribution of this work is to estimate the energy interference according to the previous knowledge of standalone application running on the same machine. They model interference as a separate implicit task. Moreover, an energy efficient co-location management policy is introduced in this work that is modeled as an optimization problem solvable by data mining techniques. All the VMs running on the same machine are known as a collection.

The energy consumption of a collection is the sum of the idle energy consumed for the longest VM run, dynamic energy consumed by each VM if they were run in isolated environment, and the energy depleted due to the interference between each VM pair. The interference energy can be positive or negative depending on the intersection of resources between each VM pair. Interference energy is estimated as the coefficient of the summation of idle and isolated run for each VM.

3.4.2 Energy non-proportional Host Effect

Increasingly, the most efficient servers nowadays, consume at least 20-30 percent of their nominal power in the idle case, and deviate from linear proportionality property noticeably according to the SPEC power benchmark[8]. Hence, Idle to Peak Ratio (IPR) and Linear Deviation Ratio (LDR), for the current power model, are still remarkably higher than the ideal case. Higher IPR encourages the server consolidation for the sake of power capping; however, this is not always a solution. Utilizing a server to its 100% capacity may affect the applications performance tremendously, thus reduces the actual energy efficiency of jobs, and also does not contribute to the power saving in cases that LDR is unequal to one and when the interference overhead exceeds the proportion of static power.

[8] `https://www.spec.org/power_ssj2008/`

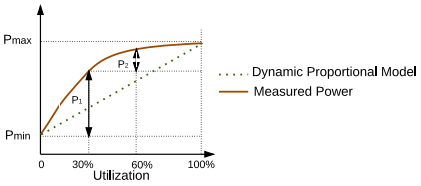


FIGURE 3.3: energy non-proportionality effect

Hence, besides the hypervisor and interference overhead in multi-tenant systems, the energy non-proportional hardware adds more complexity to the VM power modeling agenda. In energy non-proportional hardware platform, since the hardware power model is non-linear, two identical VMs, sharing the same hardware, may end up with different dynamic power usage estimation during the runtime, which may lead to an unfair energy based service charging, and planning. Figure 3.3 visualizes such a case. In this scenario, there are two identical VMs, i.e. VM_1 and VM_2, co-located in a host with the power model demonstrated in the figure. If we only run VM_1, the dynamic power estimated for this VM will be P_1, whereas running the second identical VM on the same machine predicted as $P_2 < P_1$. Therefore, in case of co-location, there should be a strategy to divide the dynamic power fairly among the running VMs.

To address the fairness issue, we propose the weighted division based VM power model. In this model as illustrated in (3.13), a particular VM's power consumption, $P_{VM}(i)$ is calculated according to the relative utilization, i.e. $\frac{u_i}{U}$, contributed by that particular VM. In this equation, u_i represents the utilization incurred by VM_i, and U denotes the overall machine utilization.

$$P_{VM}(i) = \frac{u_i P(U)}{U} \tag{3.13}$$

Moreover, this model, to some extent, considers the virtualization and interference overhead, since the VM utilization is divided by the overall system utilization. However, one should note that, higher utilization, does not necessarily mean higher virtualization overhead and interference. Virtualization overhead depends on the characteristics of the application running on it. All the same, interference overhead

depends on the characteristics of the co-located application not the overall system utilization.

3.4.3 Contention Sensitivity

While virtualization guarantees application isolation, better resource utilization and lower operational costs, they come at the price of the application slow down and inter-VM performance interference in ways that cannot be modeled or seen easily. Inter-VM performance interference happens when behavior of one VM adversely affects the performance of another due to the contention in the shared use of system resources. For example, two VMs may thrash in the shared hardware cache when running together, but fit nicely when running in isolation. Interference can happen at any level: memory, cache, I/O buffer, processor cache, etc. Co-scheduled applications may negatively interfere with each other even if they run on different processor cores because they share caches, memory channels and storage devices. Although performance interference is inevitable when resources are shared, there are ways to guarantee performance for VMs.

It is important to be able to quantify the contention experienced in a host since interference degrades the performance of applications hosted on the VMs. An application's sensitivity to contention is defined by its potential to suffer performance degradation from the interference caused by its contentious co-runners. The most trivial way to determine an application's degradation in performance is to measure the drop in execution time. However, long running user facing interactive services could use different metrics other than execution time to reflect performance. So as to cater to the general range of applications, sensitivity of an application is typically defined as [126]:

$$Sensitivity_A = \frac{IPC_A(isolated) - IPC_A(Co-run)}{IPC_A(isolated)} \tag{3.14}$$

where IPC is the instructions per cycle of the application. This metric essentially measures the drop in the number of instructions per cycle when co-located with other VMs.

Total contention overhead in a host is governed by two major sources: contention overhead and heterogeneity of resource usage due to VM co-location.

3.5 Application Power Modeling

Application energy characterization faces more challenges compared to the challenges of VM and hardware energy. Application Energy model should be accurate enough in a coarse grained view toward energy characterization and increasingly needs to be hardware oblivious. Fulfilling these requirements entails sketching a model that maps the application requirements to a set of parameters that represent the tentative resource utilization in runtime. The typical state-of-the-art approach as mentioned in the related work, is application profiling which fails to meet hardware agnosticism.

The closest work to the approach to our proposal, i.e. analytical model for application power characterization is found in [109]. However, this study is centered on the algorithms and is only studied for a limited set of algorithms. Therefore, a generalized model derived from the proposed model is required to build up a framework for algorithm energy complexity. Besides, application energy analysis in a hardware oblivious setting needs to take into account data flow among the algorithms running in processes as the basic component of an application. Note that in distributed application settings such as MapReduce, flow complexity broadens its extent to network communication.

We propose **energy complexity** to analyze the energy consumption of an application. Energy Complexity envisages energy dissipation paradigm for the application regardless of the hardware, virtualization technology, or the operating

system hosting it. ϵ represents the Least Upper Bound (LUB) in energy complexity context; whereas, the Greatest Lower Bound (GLB) is denoted by ε. To analyze energy complexity, first the application should be divided into the phases according to its different periods of resource intensiveness. Then, we can model the energy dissipation in each phase according to the associated resource power model multiplied, either by time elapsed in that phase, or the throughput of the corresponding hardware. To exemplify, in the rest of this section, we formulate storage, MapReduce and streaming service energy complexity as popular instances of the cloud services.

3.5.1 Storage as a Service

To offer storage service on a distributed system, we need a decentralized storage system e.g. Hadoop Distributed File System installed on top of the infrastructure (HDFS) [30], and Tahoe-LAFS installed in P2P-cloud instance that we have. Generally, a decentralized storage system comprises to a set of storage nodes, client nodes and coordinators. Storage nodes are coordinated by the coordinator nodes which are aware of each individual node, e.g. NameNode and Introducer in HDFS and Tahoe-LAFS.

Energy consumption of storage service factorizes to the communication, coordination and storage nodes energy dissipation.

$$E_{SaaS} = E_{coordination} + r \times (E_{communication} + E_{storage}) \tag{3.15}$$

Communication energy, $E_{communication}$ is modeled as in hardware power modeling section, i.e. Section 3.3, and $E_{coordination}$ denotes the energy consumed in the coordinator host, following the host power model formulated in the previous section. To make the data robust to failure and increase the distributed file system's reliabiliy, data is replicated in different storage nodes, by default with replication factor of 3 in HDFS and 10 in Tahoe-LAFS. r represents the replication factor in a distributed file system.

$E_{storage}$ depends on the drive technology, for the SSD drives this value is only proportional to the data size and trivially compares to the other parts energy, while for the HDD drives based on rapidly rotating technology, this energy not only depends on the data size but also data and disk head location. In HDD, power drawn for retrieving the data is not negligible. There is some effort to model disk power usage on HDD technology [9] as $E^*_{storage} = \frac{P^*_{HDD}}{\tau_{HDD}} + E_{compute}$. Where P^*_{HDD} represents the power dissipated in the hard drive for read/write operations, i.e. $* \in \{read, write\}$. Dividing P^*_{HDD} by throughput τ_{HDD}, we can calculate the energy consumption of disk for an operation. Moreover, for security reasons or to save storage, data may be stored in encrypted or compressed format. In such cases, read and write process may require additional decrypt/decompress and encrypt/compress stage, which are all computing intensive. Hence, $E_{compute}$ as the representative of these processes can be formulated using the host power model multiplied by the time elapsed to do the corresponding computing in the host, when we map the energy complexity to the energy consumption of a specific hardware. However, in the standalone energy complexity definition, the host energy for the compute phase can be defined as the algorithm energy complexity. Algorithm energy complexity is a function of the algorithm time complexity and the level of parallelism.

Therefore, energy complexity of the storage is computed as follow:

$$\begin{aligned}\epsilon_{SaaS} = {} & PUE \times P_{host} \times t + S_{data} \\ & \times [PUE \times (\sum_{h \in hops} \frac{r_h \times (P_h(host) + P_h(switch))}{\tau_h} + \frac{P^*_{HDD}}{\tau_{HDD}}) + E_{Internet}]\end{aligned} \tag{3.16}$$

If we use SSD and allocate local resources then $PUE = 1$, and do not need over the Internet communication. Therefore, ε for storage service follows (3.17).

$$\varepsilon_{SaaS} = P_{host} \times t + S_{data} \times \sum_{h \in hops} \frac{P_c(h) + P_c(s)}{\tau_h} \tag{3.17}$$

3.5.2 MapReduce as a Service

Here, we analyze the energy complexity of MapReduce applications. When a MapReduce request is sent, the scheduler decides which host should perform the job. Being assigned to the hosts, the input is split into n_t inputs of size $S_t(i)$ in the map phase. Each individual task with specified input is allocated to a host; note that more than one task may be assigned to a single host. To complete a task, a host acquires not only the task input data, but also the appropriate VM containing the execution code. Therefore, the data transmitted within the communication infrastructure may need to include, conservatively, VM and input data with size, S_{in}, where $S_{in} = \sum_{i \in n_t} S_t(i)$. In the second phase of MapReduce, i.e. the reduce phase, output is aggregated in the output file of S_{out} and delivered as the job result. Moreover, the output of the first phase, named intermediate output may be exchanged among the hosts due to the shuffle-exchange phase. Overall, the size of the transmitted data in this phase is S_{inter}. Therefore, the size of data to be transmitted is following (3.18).

$$S_d^{MR} = S_{in} + n_{hosts} \times S_{VM} + S_{inter} + S_{out} \tag{3.18}$$

S_{VM} and n_{host} denote the VM size and the number of hosts assigned to the job respectively. The output data size and intermediate output size may vary according to the MapReduce application type and the input file. The energy consumed to transmit the required data for a job can be derived applying general communication energy in (3.8). Moreover, input and output data are needed to transmit over the Internet, then this terms, $(S_{in} + S_{out}) \times E_{Internet}$, should be added to formulate the overall communication energy.

The energy drained for computing is $\sum_{i \in n_t} P_{host}(i) \times t_{task}(i)$ for each phase. t_{task} is the time to process the assigned task in the host which is directly proportional to the CPU utilization and clock frequency. In the application level, the time and space complexity of each map and reduce phase algorithms can define the energy consumed in these phases.

Therefore, the energy complexity of a MapReduce application is formulated as following:

$$\epsilon_{MR} = PUE \times [S_d^{MR} \times \frac{\sum_{h \in hops} r_h \times (P_{switch}(h) + P_{host}(h))}{\tau_c}] + \sum_{i \in phases} \sum_{j \in n_t} P_{host}(i,j) \times t_{task}(i,j) + (S_{in} + S_{out}) \times E_{Internet} \tag{3.19}$$

As we see in (3.19), the level of distribution, affects the energy usage, by using more VM images to transfer and install on the hosts. Hence, to reduce the energy consumption, a strategy would be consolidating the tasks in the minimum number of hosts as long as the resources are available and the performance is not violated. Applying this policy contributes to less VM image transmission and intermediate output data exchange. Moreover, by choosing the set of nodes within a cluster, we can save transmission energy more in the layered architecture, due to the lesser number of hops to traverse for communication.

If we serve the MapReduce applications locally, i.e. avoiding data transfer through the Internet, we can omit the last term in the MapReduce energy complexity. Moreover, by minimizing the PUE value and replication factor we can achieve the min energy complexity as:

$$\varepsilon_{MR} = S_d^{MR} \times \frac{\sum_{h \in hops} P_{switch}(h) + P_{host}(h)}{\tau_c} + \sum_{i \in phases} \sum_{j \in n_t} P_{host}(i,j) \times t_{task}(i,j) \tag{3.20}$$

3.5.3 Streaming as a Service

The video streaming service can provide either online streaming, i.e. content being encoded on the fly, or offline streaming, i.e. serving previously encoded and stored content. Thus, if we consider offline video streaming, no video rendering and encoding in the cloud side is required. Video frames are stored in the cloud storage and retrieved on demand. Video decoding, on the other hand, is always done in

the end user side. Hence, applying power aware video decoding mechanisms [105] contributes to a more energy efficient service provisioning at the end user level.

Nonetheless, in cloud assisted live video streaming, e.g. Amazon CloudFront live video streaming [9], video rendering is done in the data center servers. For instance, CloudFront uses the Adobe Flash Media live encoder.

For offline video streaming, the energy is dissipated in three parts, retrieving the frames from the distributed storage system, and transmitting them over the Internet. On the user side, this frames should be buffered and decoded to play on the screen. Since, in all scenarios, we target the same end user, we presume that the end user energy consumption is a constant amount for all given scenarios. Therefore, the energy complexity of on-demand streaming is formulated as $\epsilon^{OD}_{StaaS} = \epsilon_{SaaS} + \epsilon_u$, where ϵ_{SaaS} represents the energy complexity of storage service as disscussed before, and ϵ_u denotes the energy complexity of decoding at the end user level. All the same $\varepsilon^{OD}_{StaaS} = \varepsilon_{SaaS} + \varepsilon_u$.

For live streaming, however, we should also model the video encoding energy consumption $E_{encoding}$ and replace it with the $E_{storage}$ in (3.15). For the rest of the processes we can follow the offline streaming model. Video coding tightly couples with the video format; nevertheless, we can safely assume that the encoding energy is greater than or equal to the decoding energy in a particular hardware platform due to the exhaustive, extra stage of complicated motion compensation recognition process should be traversed in the encoding process. Here we consider H264 video format. H264 video is formed as a set of consecutive frames of three different types: I,P, and B frames. I frames are independent images while P frames are generated based on their previous I frames and B frames are coded based on the frames before and after them. Typically, the I frame coding follows the JPEG coding. B frame coding draws more power compared to the I and P frame on the same machine, since B frame relies on bi-directional differential coding of the values through the JPEG coding process, $E^I_{encoding} \leq E^P_{encoding} \leq E^B_{encoding}$. Energy complexity in live

[9] `http://docs.aws.amazon.com/AmazonCloudFront/`

streaming is formulated as the energy complexity of video rendering added to the energy required to transmit the data.

3.6 Evaluation Framework

To evaluate the introduced system, we set up a simulation and measurement framework with the configuration and scenarios explained in this section.

3.6.1 Hardware Configuration

For data center hosts we use set of Sandybridge and AMD servers. For P2P-cloud we performed our study on cloudy [10], a cloud platform on top of Confine wireless community network [11]. P2P hosts include a set of Atom and Core i7 Ivybridge computers.

Data center Network Setup Different topologies covering switch centric and server centric systems have been studied and simulated using power consumption values of switches that are available in the market. For the core switch, we opt for Cisco Nexus 5596T, which has 32 ports of 10 Gbit Ethernet, and supports optical networking due to SPF+ ports. It typically dissipates 900 W, while the maximum power is 1100 W.

For the distribution and access layer switches, we rely on the Cisco Nexus 2232TM switch, which has 32 ports of 1 Gbit and 10 Gbit Ethernet, with the over subscription of 4:1. Its maximum power consumption is 386 W; nonetheless, it draws 280-350 W, typically. For commodity switches, we employ Cisco Catalyst 37590-48TS, which consumes the maximum of 75 W and provides 48 ports. For all the switches, because they include the recent technology of green switches, the power

[10] http://cloudy.community/download/
[11] http://confine-project.eu/

TABLE 3.1: Data center network configuration

Layer	Switch	number of ports	Communication type	Power-average	Power-max
Core	Cisco Nexus 5596T	32	10Gbit Ethernet	900W	1100W
Distribution	Cisco Nexus 2232TM	32	1 & 10Gbit Ethernet	300W	386W
Access	Cisco Catalyst37590-48TS	48	100Mb & 1Gbit Ethernet	66W	75W

drawn for each port, in idle case, is almost zero. The data center network setup is summarized in Table 3.1.

N.B: All the above values are derived from the devices' datasheet. Table 3.1 summarises the above values. All the data is derived from devices data sheets. For all the switches, since they include the recent technology of green switches, the power drawn for each port, in idle case is almost zero. The power consumption of an active server port is set to be 3W [58]. Exploiting Gigabit Ethernet, data center network performance is more than 90%; therefore, τ_{DC} is above 967 Mbps.

Internet Power Consumption Internet energy consumption values are derived from [10], which characterizes the distribution and access power of around 10.25 W and core power of less than 0.15 W per connection through a fast Ethernet link, with the over subscription of 40:1.

P2P Communication Setting We characterize the power consumption in the P2P-cloud by means of experimental measurements in a production wireless community network. The network consists of around 50 802.11an-based nodes. It is deployed as part of the *Quick Mesh Project* (QMP) [12] and EU CONFINE project [13]. We refer to this network as *QMPSU*, which is part of a larger Community Network having more than 20.000 operative nodes called Guifi.net [14]. An experimental evaluation of QMPSU can be found in [34], and a monitoring page is available in the Internet [15].

[12] `http://qmp.cat`
[13] `http://confine-project.eu/`
[14] `http://guifi.net/en`
[15] `http://dsg.ac.upc.edu/qmpsu`

TABLE 3.2: HDD Specifications

Scenario	Type	Throughput random	Throughput sequential	P_{idle}	$P_{sequentialread}$	$P_{randomread}$
Data center	Western Digital RE NAS 4TB	6 Gbps	182 Mbps	8.9 w	10.2 w	10.9 w
P2P-cloud	WD Scorpi Blue (WD5000BPVT) 500MB	3 Gbps	136 Mbps	0.65 w	1.6 w	–

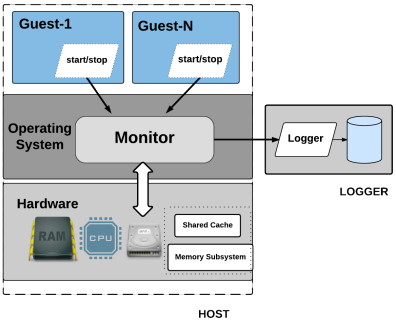


FIGURE 3.4: Monitoring infrastructure

Hard Disk Characterization For the storage in our evaluation, we follow the hardware specified in Table 3.2.

3.6.2 Monitoring Setup

We have developed a monitoring infrastructure to collect several system-related metrics during the execution of the workloads in our measurements [16]. Figure 3.4 presents the elements that constitute this system. The main monitoring component is located at the host, where one or more guest VMs are running. This component collects four types of data: i) Frequency at which each of the cores is running; ii) Performance counters, including the number of processing cycles, the number of instructions, the references and misses when accessing the cache and page faults of the virtual memory; iii) Overall CPU usage, memory usage, disk reads and writes; iv) Instantaneous power consumption by reading an external power meter attached to the host.

[16] Monitoring tool is available online at:
`https://github.com/lsharifi/utility-monitor`

Guests are responsible for the life cycle of monitoring activities. When a new experiment starts at the guests, a notification is sent to the host, requesting a specific type of monitor to start. The guest also notifies the host to stop a given monitor when the experiment finishes. The host collects the requested information and redirects it to a *logger* machine, to minimize interference with disk operations. Communication between all parties is socket-based.

3.6.3 Experiment Scenarios

We elaborate the energy consumption in P2P-cloud and data center under the same workload condition. We aim to analyze the energy consumption on different cloud models under the storage service, MapReduce workload, video compression, ray-tracing and some scientific computing workload with the following configuration.

Storage Service Setting: The setting related to a storage service is based on distributed transfer of files. We consider the Secure Copy (SCP) to transfer files between different nodes. The setup is for transferring an ISO from the Cloudy distribution, duplicated 4 times, making a total file transfer of 1.35 GB [17]. We have two scenarios, one that is read-dominant (copy from VM to another host) and one that is write-dominant (copy from another host to the VM).

MapReduce Setting: In this part of the evaluation we have used the Tera-Sort and WordCount applications. These applications explore the architecture of MapReduce and are within its common operations. TeraSort works upon sequences of records of 100 bytes in length, where the first 10 bytes are a random key. To generate such a sequence we used a companion tool, the TeraGen, which generates a number of 100 byte records.

Regarding WordCount, the mapping phase splits the input into tokens and assigns them an initial counting value of 1, with each mapper receiving an individual line of the file. In the reduce phase, the words are aggregated and counted. Both

[17] `http://cloudy.community/download/`

applications use the same input, generated by TeraGen, i.e the overal data size of 1 GiByte.

Video Compression: Video compression is a computation heavy application and is becoming one of the common cloud services. For this, we use h264ref application from SPECcpu2006 benchmark suit [18]. 464.h264ref is a reference implementation of H.264/AVC (Advanced Video Coding), the latest state-of-the-art video compression standard. The standard is being applied for applications such as the next-generation DVDs (Blu-ray and HD DVD) and video broadcasting. The 464.h264ref source code is based on version 9.3 of the h264avc reference implementation.

Ray-tracing: Ray-tracing is a rendering technique that calculates a photo-realistic image of a three-dimensional scene by simulating the way rays of light travel in the real world but it does so backwards. For ray-tracing, we use 453.Povray from SPECcpu2006 benchmark suit. Povray algorithms are mostly sensitive to floating-point computations.

Scientific computing: For scientific computing, we explore lbm, libquantum and namd from SPECcpu2006 benchmark suit. **lbm** implements the so-called "Lattice Boltzmann Method" (LBM) to simulate incompressible fluids in 3D as described in [106].

libquantum is a library included in SPECcpu2006 benchmark for the simulation of a quantum computer. Quantum computers are based on the principles of quantum mechanics and can solve certain computationally hard tasks in polynomial time. It has also been deployed to analyze quantum cryptography.

namd in SPECcpu2006 is derived from the data layout and inner loop of NAMD, a parallel program for the simulation of large biomolecular systems.

[18] `https://www.spec.org/cpu2006/`

3.7 Results

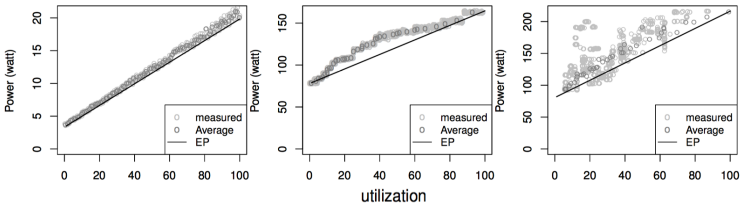
In this section, we experimentally elaborate on the concept described so far in this chapter. Here, first we sketch a power model for the hardware we introduced as our experiment platform in the previous section, and characterize their energy proportionality. Then, VM level power modeling and the effect of multi-tenancy and virtualization technology are studied.

3.7.1 Host Power Modeling

To make a power model for the hosts in our measurement testbed, we rely on two methods. First we do measurements using an external power meter attached to the devices. Secondly, we make a model based on correlating the performance counter values to the power consumption of the devices, as explained in the previous section. To summarize, the following steps are taken to make a counter based power model. 1) Specific set of microbenchmarks designed to stress each of the machine's components (i.e. CPU, memory, and I/O) at different levels of utilization. The microbenchmark is designed using *stress* tool[19], combined with CPULimit tool[20] (Microbenchmark is available online[21]). 2) The microbenchmarks are executed to gather the required data to train the model. 3) The model is produced by applying incrementally linear regression techniques on the training data for deriving the weight in power consumption of each power component defined.

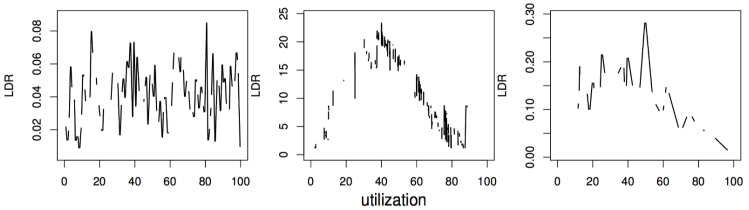
For power sampling in P2P-cloud nodes, we use Watts up pro power meter[22] and for the data center, an HP Intelligent Power Distribution Unit (iPDU)[23], with accuracy of 1.5W at current more than 20 mA is attached to the hosts and the power data is available through a Ganglia monitoring system[24]. Note that we

[19] `http://linux.die.net/man/1/stress`
[20] `https://github.com/opsengine/cpulimit`
[21] `https://github.com/lsharifi/Stress-benchmark`
[22] `https://www.wattsupmeters.com/secure/products.php?pn=0&wai=211&spec=4`
[23] `http://www8.hp.com/h20195/v2/GetDocument.aspx?docname=c04123329`
[24] `http://bscgrid28.bsc.es/ganglia2/`



a) ATOM b) Core i7 c) Xeon server

FIGURE 3.5: Power model of studied hosts



a) ATOM b) Core i7 c) Xeon server

FIGURE 3.6: LDR for studied hosts

disabled hyperthreading and turboboost all through our experiments and pinned the frequency to the range of 1.9-2 GHz.

As shown in Figure 3.5-a the power dissipated in ATOM machines varies in range of 3.5W and 20W, while in Corei7 device, Figure 3.5-b, it is in the range of 80.67W to 216.3W. Data center Xeon hosts dissipate minimum of 74W and maximum of 216W, as demonstrated in Figure 3.5-c.

3.7.2 Energy Proportionality

Figure 3.5 and Figure 3.6 characterize the power usage and corresponding LDR for all the hosts we used in our study, respectively.

As we see from the LDR values in Figure 3.6, Core i7 and Xeon CPUs are more energy proportional when they are either underutilized or utilized for the 100%. The worst LDR takes place around 50% utilization. Although, intuitively, the higher utilization leads to less cost, from energy efficiency vantage point, this may

FIGURE 3.7: QMPSU connectivity

not be true in a certain range of utilization. Thus, when deciding on the co-location on our studied servers, we should take this factor into account to prevent the range of 30-50% utilization, to be more energy efficient. However, as we see in Figure 3.6, for low power ATOM devices, the LDR variation is less than 0.08, and negligible, hence.

3.7.3 P2P Communication Power Model

Typically, QMPSU users have an outdoor router with a wifi interface on the roof, which establishes wireless links with other users in the neighborhood. Additionally, the outdoor router has an Ethernet interface connected to an indoor AP as premises network as depicted in Figure 3.7.

From the QMPSU graph formed by the outdoor routers, we obtained an average path length of 3.78 hops, thus, crossing 4.78 outdoor routers. Therefore, the average power consumption of transmission between a pair of nodes in the network is:

$$P_{WN} = 2\,P_{\text{AP}} + 4.78\,P_{\text{router}}, \tag{3.21}$$

where P_{AP} and P_{router} are the power consumption of the AP and outdoor routers, respectively.

Additionally, experimental measurements show an average throughput of 10.9 Mbps between nodes and their gateway (see [34]). This can also be estimated as the average throughput between any pair of nodes. Regarding the round trip time (RTT), experimental measurements in QMPSU give an average end-to-end RTT of 18.3 ms, with standard deviation of $\sigma = 50.6$ ms.

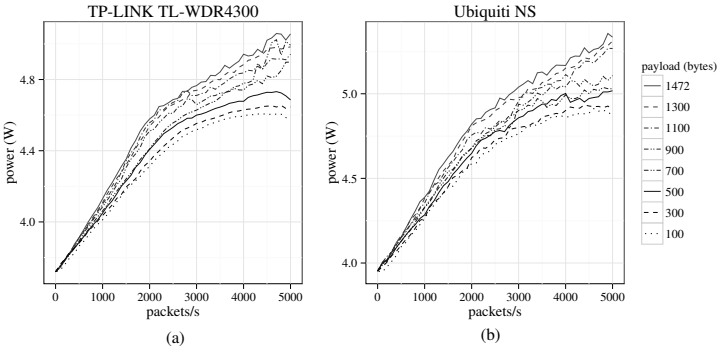


FIGURE 3.8: Power consumption of a typical indoor AP and outdoor router with UDP traffic.

TABLE 3.3: Wireless Infrastructure Power Consumption

	Power NS	**Power TP-LINK**
Static	3.7	3.9
UDP-Max throughput	5.0	5.4
TCP-Max throughput	5.2	6.1
$throughput = 10.9$ Mbps	5	5

The most common outdoor router used in QMPSU is the Ubiquiti NanoStation M5 (NS) [25]. As an indoor AP, we considered the TP-LINK WDR4300 [26]. For the power measurements we relied on our own hand-made Power over Ethernet (PoE) meter [27]. We used a voltage and current sensors, and took power samples using an Arduino type board.

Figure 3.8 illustrates the power consumption measured at the (a) TP-LINK and (b) NS devices using UDP traffic vs. packets per second, varying the UDP packet payload. Each point in the figure is obtained averaging around 5×10^4 power samples captured during 1 minute, as described above. Figure 3.8 shows static power of 3.7W and 3.9W, and maximum power of 5W and 5.4W, for the TP-LINK and NS, respectively. A TCP experiment is performed, as well. We obtained a throughput of 65 Mbps and power consumption of 5.2W and 6.1W at the TP-LINK and NS, respectively. The measured values of P_{router} for different hardware in QMPSU is given in Table 3.3.

[25] `http://www.ubnt.com/downloads/datasheets/nanostationm/nsm_ds_web.pdf`
[26] `http://www.tp-link.com/lk/products/details/?model=TL-WDR4300`
[27] `http://dsg.ac.upc.edu/poe_power_meter`

3.7.4 VM Power Modeling

Here, we use KVM [28] and QEMU [29] as the virtualization infrastructure. QEMU is a generic and opensource machine emulator and virtualizer that achieves near native performances by executing the guest code directly on the host CPU using KVM kernel module. The applications run in the VMs are Memcached [30] and MBW [31]. Memcached is a widely used in-memory storage system and MBW is a memory intensive application that is used to compute the memory bandwidth of a machine. Both these applications contend for Last Level Cache (LLC) and the Memory Controller. This contention degrades the performance of both applications and causes some changes in instantaneous power dissipation pattern.

3.7.4.0.1 VM power model accuracy and energy non-proportionality effect: It is not possible to validate the per-VM estimations against empirical data, because the current servers only provide aggregated power values. Thus, to validate our VM energy accounting, we need to assume that the sum of the predicted energy consumption of each virtual machine running on the system must be the same as the overall platform's energy consumption. Even when these values are not actually the same, we consider the error to be uniform (i.e., proportionally the same difference) across VMs.

Following the above validation method, in Figure 3.9, considering the energy non-proportionality of the servers - i.e. bars labeled as enp-* - the cumulative power dissipation of the co-located VMs is very close to the total power drawn in the server. However, in this figure, ep bars which ignore the effect of multi-tenancy in the energy non-proportional ecosystems, demonstrate a huge margin of error in the VM power estimation, hence clearly justifying the need for our proposed model.

[28] `http://www.linux-kvm.org/page/Main_Page`
[29] `http://wiki.qemu.org/Main_Page`
[30] `http://memcached.org/`
[31] `http://manpages.ubuntu.com/manpages/utopic/man1/mbw.1.html`

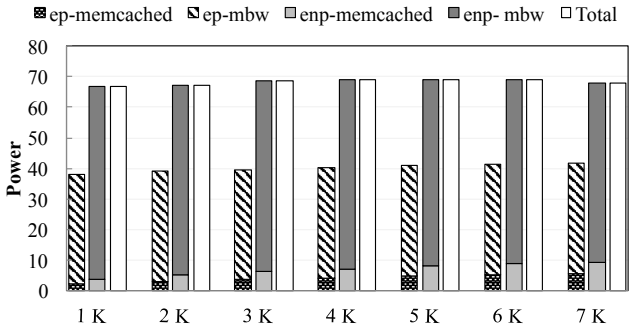
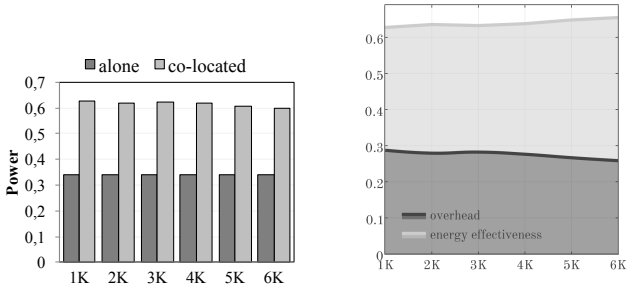


FIGURE 3.9: Effect of energy non-proportionality in VM power modeling. X-axis shows the workload for Memcached in Requests Per Second (RPS) and Y-axis shows the normalized power

3.7.4.0.2 Interference Effect: Figure 3.10 shows the power consumption of MBW VM instance, running on six cores, alone and also co-located with Memcached with different workloads. From Figure 3.10-a, we see that for most scenarios, power dissipation increases up to 1.5% in co-located runs. Since the execution time is not a valid metric for MBW, we cannot directly perform energy accounting for this application. Nonetheless, considering a long runtime period, ~1% increase is mapped to a remarkable energy amount.

Moreover, for those classes of applications, which terminate after a while, resource utilization may not increase or even slightly decrease in one sample. However, overall energy consumption, which represents the cumulative value of the samples, increases as a result of longer execution time. Therefore, scheduling algorithms should be revisited, baring this fact into account.

In cases that static power division, among the VMs, cannot compensate the co-location overhead, multi-tenancy fails to reach energy capping goals. Besides, as mentioned in the definition of *Energy Effectiveness* in Section 3.2, energy and performance should be considered together in any scheduling decision. For instance, co-locating Memcached with MBW significantly increases the latency of Memcached. Figure 3.10 depicts the overhead of co-location as well as energy effectiveness for the above scenarios.



a) Power Drawn b) Co-location Overhead

FIGURE 3.10: Power Consumption of MBW

3.7.5 Virtualization Technology

Figure 3.11 shows the average energy consumption of running the applications described in Section 4.3.2, either *i)* directly on the native host operating system, or *ii)* by using Linux containers (LXC), and *iii)* the kernel-based virtualization (KVM). In this experiment applications are executed inside each virtualization container. The containers were limited to either 1 or 2 virtual cores, which are pinned to physical cores.

As we see in this figure, different VMs suit different applications. For instance, in single thread applications, such as H264ref, increasing the number of cores, may lead to increase in energy consumption, due to more active cores and little improvement in the performance. Moreover, in all cases LXC dissipates less energy compared to the KVM, thanks to its near native execution, which happens due to sharing the kernel/OS code.

Figure 3.13 shows the results for energy effectiveness, our service level evaluation metric in the framework, as explained in Section 3.2.

The advantage of running LXC in the P2P-cloud is demonstrated by these sets of results as shown in Figure 3.11, 3.12, and 3.13. LXC is currently deployed on Cloudy platform [32] (a Debian-based Linux distribution also targeting Raspberry

[32] `http://cloudy.community/`

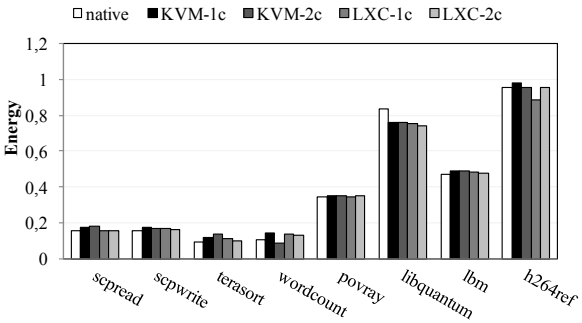


FIGURE 3.11: Effect of virtualization technology on energy

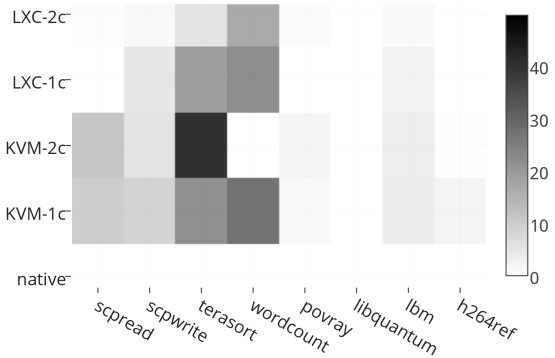


FIGURE 3.12: Virtualization Overhead

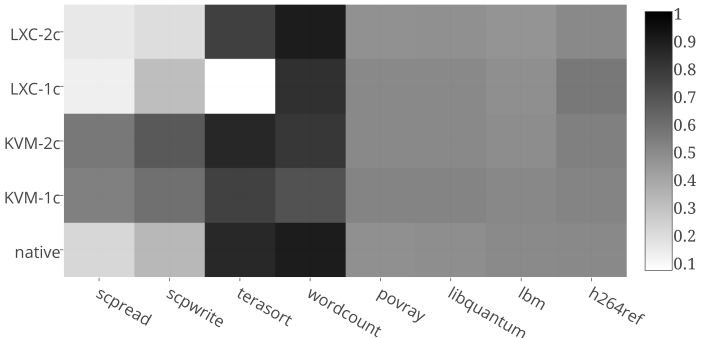
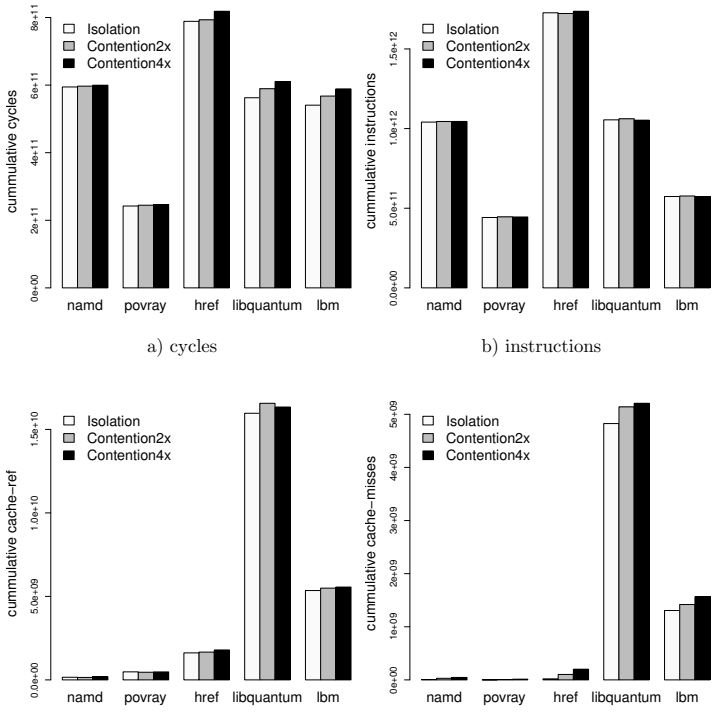


FIGURE 3.13: Energy effectiveness vs. virtualization technology



a) cycles b) instructions

c) cache references d) cache misses

FIGURE 3.14: Interference effect on CPU and memory intensive workload

Pi devices), which is a community network based cloud. KVM leads to an extra energy consumption, in average, when compared with LXC.

3.7.6 Virtualization Density

To quantify the effect of virtualization density and co-location on application energy consumption and energy effectiveness, we study a set of workloads from SPEC-CPU2006 benchmark [33], including: lbm, libquantum, povray and namd. The first two are memory and CPU intensive, while the latter ones are only CPU

[33] `https://www.spec.org/cpu2006/`

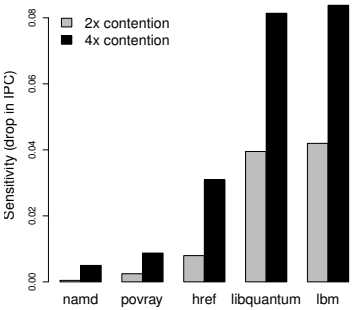


FIGURE 3.15: Sensitivity of different workloads when co-located with gcc

intensive. We draw the cumulative value of four generic counters in Figure 3.14, i.e. cycles, instructions, cache references and cache misses. The difference in the cumulative value of three counters that we employed in our power model, indicates the excess energy consumption due to the co-location excluding the static power dissipated, while the cumulative cycles difference is the indicator of performance degradation due to interference and co-location overhead. Also, longer residual time causes for more of static power consumption to be charged for a specific application. Therefore, co-location is worthy just if the following condition is met:

$$\frac{p_s \times (\Delta t + t_{isolated})}{N_{VMs}} + \sum_{c \in counterset} \Delta CF(c) < p_s \times t_{isolated}$$

This statement implies that the excess power induced by co-location and interference should be less than the static power drawn during an isolated run. Note that $\Delta*$ represents the difference of values for parameter $*$ in co-located and isolated runs; that $CF(c)$ denotes the cumulative value of counter c at the end of a run; and that counter set includes: instructions, cache references and cache misses.

Figure 3.15 depicts the sensitivity of the different SPEC benchmarks to contention with another workload, gcc (SPEC-CPU2006) as explained in Section 3.4.3. We chose gcc from SPECcpu2006 benchmark suit for co-location with all our studied workloads due to its longer execution time, which guarantees interference during

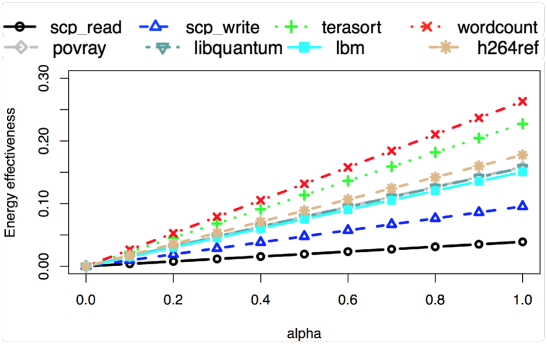


FIGURE 3.16: Vulnerability Factor

the whole run cycle of our selected workloads. Since povray and namd are CPU intensive, they have much lesser sensitivity to contention in comparison with lbm and libquantum. This observation has a direct consequence and can aid in increasing the virtualization density in cloud machines. Co-locating CPU intensive applications with Memory intensive applications can, thus, result in a higher VM density without actually degrading performance, and accruing excessive penalty in energy consumption.

3.7.7 Energy Effectiveness and Vulnerability Factor

In Figure 3.16, sensitivity of energy effectiveness to α parameter tuning, i.e. vulnerability factor, is demonstrated. This figure indicates how important is the tuning of the α parameter for different applications. Figure 3.16 depicts that applications sharing the same general characteristics show similar behavior to α tuning. Terasort and Wordcount, on top, are both MapReduce applications, which are computing and I/O intensive applications. H264, lbm, libquantum and povray as computing intensive applications follow, almost, the same pattern by varying the α value. However, SCP-read/write on the bottom of the graph, as I/O intensive applications, indicate the least range of variation due to α changing.

Moreover, this figure reveals that the computing intensive applications are more vulnerable than communication intensive ones to the α miss-configuration. This phenomena stems from the fact that computing intensive applications are more energy hungry and sensitive to the performance drop.

3.8 Summary

In this chapter, as the major contribution, we addressed the formulation of a performance aware energy analysis metric by introducing energy effectiveness, which can be specified in each layer of service provisioning stack, i.e. application, virtual machine/OS and hardware, from a course-grained, asymptotic, hardware agnostic conceptualization on top of the stack to an accurate, fine-grained, hardware dependent formulation on the bottom layer.

Further to proposing energy effectiveness, we introduced an approach to estimate energy across the stack. We formulated power consumption in communication and process elements of different cloud platforms, and discussed the added complexity in power modeling rooted in the multi-tenancy as the cornerstone of cloud service provisioning. Besides, we characterized the energy complexity in the application layer via three examples of Storage as a Service, Streaming as a Service and MapReduce.

In the next chapter, we form a service analysis framework, leveraging the energy effectiveness metric we introduced here, to improve energy effectiveness by efficient service platform selection.

—"Your scheme must be the framework of the universe;
all other schemes will soon be ruins."

-Henry David Thoreau

4

Analysis Framework

P2P-clouds consist of vast sums of ubiquitous commodity ICT resources, which introduce an opportunity to scale the cloud service provisioning beyond the borders of giant cloud service providers such as Amazon that rely on gigantic data centers. However, since energy consumption is becoming crucial in the industrial world, including IT sector, emerging technologies should be energy efficient enough to be able to survive in the new economics paradigm.

To understand if P2P-cloud, as an emerging cloud paradigm meets the above condition, in this chapter we sketch a comprehensive view of energy consumption within a service life cycle. A hardware agnostic framework is needed to cope with the hardware diversity. It is required to assess, or provide meaningful estimation of the energy consumption on any hardware platform. Leveraging such a framework

assists the resource manager module and broker to make energy aware decisions for resource allocation in the federated environment of P2P-clouds and data centers.

As the major contribution in this chapter, after surveying the related work in energy aware scheduling and resource allocation in Section 4.1, we introduce an energy analysis framework to compare the effectiveness of a diverse range of services in P2P and data centers in Section 4.2. We scrutinize the energy consumption for the MapReduce applications as a case study in Section 4.3. Afterwards, using our framework, we compare different cloud services in both data center and P2P-cloud scenarios, keeping data center granularity in perspective, Section 4.4. Relevant publications to this chapter's contributions are [113, 118].

4.1 Related Work

Power management in data centers embodies in local and efficient electricity source selection, thermal management, workload consolidation and task scheduling [98]. Dynamic power management is usually translated to server consolidation which includes VM migration and server shutdown or hibernation [99].

4.1.1 Server Consolidation

Server consolidation is one of the most promising energy oriented scheduling solutions which offers dynamic capacity provisioning through turning on and off the data center servers [16, 17, 143, 146]. Beside energy capping in this technique, several aspects including service performance and reliability should be factored in the scheduling mechanism.

In [20] a framework is proposed for intelligent consolidation using different approaches such as turning on/off machines, power-aware consolidation algorithms, and machine learning techniques to deal with uncertain information while maximizing performance. They employed models learned from the previous system

behaviors to predict power consumption levels, CPU loads, and SLA timings, in scheduling decisions.

Aiming at server consolidation, research in [143] investigates the resource provisioning problem from the cloud provider's perspective, where resource demand and usage are multi-dimensional. This solution considers resource usage and capacity for multiple resource types, such as CPU and memory.

4.1.1.0.1 VM migration: In tandem with server consolidation, VM migration is introduced as a key solution. Beloglazov et al.[16] present a decentralized architecture of a resource management system for cloud data centers that aims to use continuous optimization policies of VM placement. They look at factors such as CPU, RAM, network bandwidth utilisation, and physical machine's temperature, to better reallocate machines and improve overall efficiency. In their other work [17], they detect over-utilisation and under-utilisation peaks to migrate VMs between hosts and minimize the power consumption in the data center.

Research in [17, 55, 66, 139] has relied on full VM migration to reduce energy consumption by consolidating VMs and switching hosts to low-power mode. Nonetheless, full VM migration requires the target host to have enough resource slack to accommodate the incoming VMs, resulting in low consolidation ratios. In contrast, Oasis [146] implements a hybrid approach that uses partial and full VM migration to achieve very high consolidation ratios and save energy in heterogeneous server architectures. Partial VM migration consolidates only the working set of idle VMs and lets VMs fetch their memory pages on-demand [24].

4.1.1.0.2 Dealing with the latency issue: Server consolidation may increase latency, due to boot up delay [99]. To tackle this issue, PowerNap [90] describes a mechanism to eliminate idle power waste by letting servers quick transition between high and low-power modes in response to workloads. Moreover, Isci et al. [66] describe a virtual server platform that supports low-latency, low-power modes.

4.1.2 Tackling System Dynamics

The major challenges in energy efficient resource allocation are composed of choosing workload type and interference between different workloads, and resource usage, performance and power consumption. Tangential with this trend, adaptive resource allocation policy design is crucial. This dimension refers to the degree to which an energy aware resource allocator is able to adapt to dynamic or uncertain conditions. The uncertainty arise from a number of factors including: resource capacity demand, failures and user workload pattern [60].

The lack of energy proportionality of typical computer hardware and the fact that important workloads (such as search) require all servers to remain up regardless of traffic intensity renders existing power management techniques ineffective at reducing energy use. PEGASUS solution [83] presents a feedback-based controller by exploiting request latency statistics to dynamically adjust server power management limits in fine-grain, running each server just fast enough to meet global service-level latency objectives.

Two main optimization approaches have been proposed for energy-aware scheduling in the state-of-the-art. The first is the independent approach, which assumes that energy and performance are independent goals for the optimization problem. The second is the simultaneous approach, which optimizes performance and energy at the same time, modeling the problem as a multi-objective optimization. Of these two, the simultaneous approach is the most comprehensive, because the algorithms are oriented to find Pareto optimal schedules. Therefore, no single scheduling decision can strictly dominate the others with better performance and lower energy consumption at the same time.

Lee and Zomaya [79] studied several dynamic voltage scaling (DVS) based heuristics to minimize the weighted sum of makespan and energy. A makespan-conservative local search technique is used to slightly modify scheduling decisions when they do not increase energy consumption for executing jobs, in order to escape from local optima. Mezmaz et al. [91] improved on [79] by proposing a parallel, bi-objective

hybrid genetic algorithm for the same problem using the cooperative distributed island/multi-start model, which significantly reduces the execution time of the scheduling method.

Pinel et al. [102] proposed a double minimization approach for scheduling independent tasks on grids with energy considerations, first applying a heuristic approach to optimize makespan, and then a local search to minimize energy consumption. They proposed greedy and genetic algorithms to solve the makespan-energy scheduling problem subject to deadline and memory requirements.

Energy-aware control techniques have also been developed for networks of data centers. Work in [95] introduced an energy consumption model for multi-core computing systems. The new model is based on the energy required by the system to operate at full capacity, the energy when not all the available cores of the machine are used, and the energy that each machine on the system consumes in idle state (MIN-MAX mode).

Work in [96] presents a multi-objective optimization approach applied to the problem of operating a data center while taking into account power profiles, temperature, and QoS (modeled by task deadlines). The data center model considers a computing infrastructure, heating ventilation-air conditioning and free cooling systems, and renewable power sources. A fully multi-objective approach is used for the data center planning problem, which is solved following the simultaneous approach for scheduling. All three problem objectives are simultaneously optimized. The proposed schedulers provide a set of non-dominated Pareto solutions in each run, which account for different trade-off solutions to the data center planning problem, to be used by the data center operator in different situations.

EQVMP (Energy-efficient and QoS-aware Virtual Machine Placement) [136] is a solution with three objectives, inter-machine communication and energy reduction with load balancing. They group the machines to reduce communication and the allocation is done by finding the machine with the resource availability closer to the request. By controlling the information flow, they manage to migrate VMs and keep improving the disposition of the VMs.

PreAntPolicy [44] consists of a prediction model based on fractal mathematics and a scheduler on the basis of an improved ant colony algorithm. The prediction model determines whether to trigger the execution of the scheduler by virtue of load trend prediction, and the scheduler is responsible for resource scheduling while minimizing energy consumption under the premise of guaranteeing the Quality-of-Service (QoS). This approach offers an effective dynamic capacity provisioning model for resource-intensive applications in a heterogeneous computing environment and can reduce the consumption of system resources and energy when scheduling is triggered by instantaneous peak loads.

FORTE (Flow Optimization based framework for request-Routing and Traffic Engineering) [49] dynamically controls the fraction of user traffic directed to each data center, in distributed data center model, in response to changes in both request workload and carbon footprint. It allows an operator to navigate the three way trade-off between access latency, carbon footprint, and electricity costs and to determine an optimal data center upgrade plan in response to increases in the traffic load. Authors in [49] show that carbon taxes or credits are impractical in incentivizing carbon output reduction by providers of large-scale Internet applications.

The state-of-the-art approaches aim to reduce energy consumption and improve resource usage, but they only consider scheduling within data centers, without addressing resource scheduling in wider areas such as vicinities in P2P-clouds, and, as such, they: i) do not consider energy cost of networking in wide area, ii) thus, they only provide simplified modeling of energy consumption of workloads, iii) do not help in the problem of deciding whether to schedule in data center or in P2P-cloud, iv) they do not fully incorporate any notion akin to energy effectiveness, which adaptively combines performance and energy conservativeness in scheduling decisions.

To the best of our knowledge, there is limited work addresses the energy consumption analysis in P2P platforms. In [94] a high level model of P2P and data center

energy consumption is introduced, and [130] compared streaming services in nano-data centers with gigantic ones in terms of energy consumption. In this chapter we introduce an analytical framework to characterize service energy consumption in a P2P assisted cloud platform.

The work described in this chapter [113, 117] reveal that, in the contest between classic data centers and P2P-clouds, the latter can compete with the classic data center model in terms of energy efficiency for specific services, as long as the jobs are served mostly locally. Nonetheless, there is no straightforward global answer for this question, since energy consumption depends on a diverse range of factors on service provisioning stack, from hardware specifications to the service characteristics and execution platform.

4.2 Shaping an Energy Efficient Overlay

Since resource scarcity is a challenge in P2P-cloud, one may think that increasing the number of resources can solve the issue. However, more resource availability incurs more static power consumption in the system and in case of wireless communication, interference may occur. Therefore, there is a trade off between resource availability and energy efficiency. Nonetheless, there are some mechanisms that can improve the service experience in P2P platforms without increasing resource availability such as caching. Equipping each vicinity with local cache can decrease the communication required through the Internet which is the most expensive communication in terms of energy efficiency [117]. We evaluate the effect of using local cache in P2P in Section 4.4.

Furthermore, leveraging P2P-cloud, heavily decentralized, brings two advantages: i) cooling becomes a non-issue compared to data centers, with energy and infrastructure savings, and ii) the lower and more scattered energy usage lowers the peak power requirements, thus increasing the chances of obtaining energy from

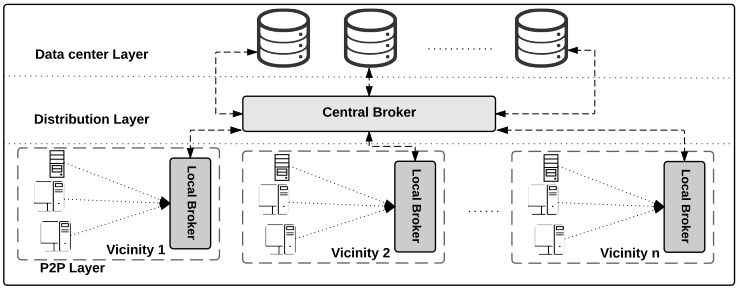


FIGURE 4.1: P2P Assisted Cloud Architecture

renewable, local or retail energy, without requiring the additional massive investments by providers in wind, solar, etc. power plants to ensure availability during such peaks.

Nevertheless, since P2P cannot fulfill all the performance requirements of every service, a federated platform of P2P and data center, as illustrated in Figure 4.1, can be a solution to improve the energy effectiveness of service provisioning in a multi-tenant platform. As shown in Figure 4.1, P2P nodes contribute to service provisioning through local broker, which prioritizes the local resources in allocation process. Nonetheless, if the local resources, within the vicinity, cannot fulfill the service requirements, local broker contacts central broker to provide the resources through data centers.

4.2.1 Energy Aware Platform Selection

Generally, each service consumes energy on hosts and communication infrastructure. Host energy factors in CPU, memory, storage and other I/O devices energy. Hence, by analyzing each service to understand the hotspots in resource consumption, we can derive a conclusion about which platform better suits the service.

Equipped with an insight into the energy consumption pattern of each individual service, local brokers can make energy-aware service platform selection and resource allocation in the federated cloud architecture, as shown in Figure 4.1.

Algorithm 1 Resource management in broker

1: **function** RESOURCE_MANAGEMENT($host\_power\{cpu, disk, memory\}$, $host\_qos\{mips, disk\_r/w\_throughput\}$)
2: Insert(*host* , ascending sorted *hostlist* according to cpu power)
3: Insert(*host* , ascending sorted *hostlist* according to memory power)
4: Insert(*host*, ascending sorted *hostlist* according to disk power)
5: **end function**

In this layered architecture, local brokers are responsible for collecting *power* and *capacity* of the hosts in the edge layer. Local broker holds an ascending sorted host list, sorted based on different characteristics of host; thus, every time a new peer wants to join the system, its characteristics are added to this sorted list, as explained in Algorithm 1.

Therefore, considering Binary-tree implementation, the complexity of updating and maintaining this list is $O(\log n)$, where n represents the number of peers in the vicinity. Besides, local brokers are connected to the central broker which holds the same information for data centers and follows the same procedure to keep the list updated. However, this list is less dynamic compared to the one in the local broker.

In Algorithm 2, we portray how a broker can take into account the energy awareness in resource allocation process for each service. In this algorithm, when the energy consumption for the service in P2P is more than the most energy conservative data center, the service is directed to be served in the data center (i.e. line 7). Otherwise, according to the service characteristics, either it is CPU-intensive, memory-intensive or I/O-intensive (computable through $analyze\_service(service)$ function). Application intensiveness, in this function can be decided according to the minimum hardware requirements to run it. Then, the energy effectiveness of the most energy conservative resources of P2P resources required to serve the service is calculated. If this value meets the minimum service quality and the resources are available, service is directed to the P2P (i.e. line 10); otherwise, it has to be directed to the data center.

The core of this algorithm is implied in line 10; where, the decision should be

Algorithm 2 Resouce allocation policy

1: **function** RESOURCE_ALLOCATION($hostlist, service, \alpha$) ▷ α represents the effectiveness factor
2: intensiveness=analyze_service($service$) ▷ analyze_service returns either of these values {cpu,memory,disk}
3: i=0
4: **while** ($i < length(hostlist)$) **do**
5: $\Delta E_{compute} \leftarrow \sum_{i \in N^{P2P}_{host}} E^{P2P}_{hostlist[intensiveness][i]}(U) - \sum_{i \in N^{DC}_{host}} E^{DC}_{hostlist[intensiveness,top]}(U)$
6: $\Delta E_{communic} \leftarrow N^{P2P}_{hops} \times E^{P2P}_{communic} - N^{DC}_{hops} \times E^{DC}_{communic}$
7: **if** ($\Delta E_{compute} + \Delta E_{communic} \geq 0$) **then**
8: **return** Direct to data center
9: **end if**
10: **if** ($\mathfrak{E}$(hostlist[intensiveness][i]))$\geq \mathfrak{E}$(*) and (resource available)) **then**
11: **return** allocate($hostlist[intensiveness][i]$))
12: **else**
13: increment(i)
14: **end if**
15: **end while**
16: **return** Direct to data center
17: **end function**

made by calculating the energy effectiveness of running the application on the host appearing on top of the *hostlist*. Therefore, the average and worst case time complexity of our resource allocation policy algorithm is $O(n)$; where, n denotes the number of nodes in the vicinity. This order of complexity indicates that more resource availability in the vicinity not only increases the static power, but also slows down the resource allocation phase.

As a result, defining the right vicinity size plays a leading role in moving toward energy efficient P2P-assisted cloud platform. To define each vicinity borders, diverse range of mechanisms have been proposed [138, 142, 144]. For instance, location awareness is attainable via techniques such as using RTT [138, 144], Euclidean distance [145], longest IP prefix matching and AS (Autonomous System) numbers [142]. However, the optimum number of nodes in the vicinity, keeping energy efficiency in perspective, is still an open question.

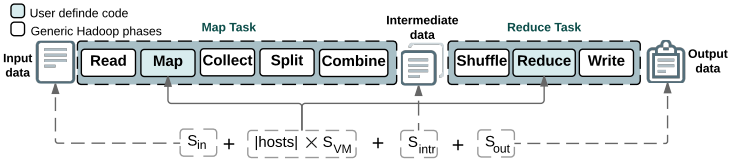


FIGURE 4.2: MapReduce Data Flow

4.3 MapReduce Case Study

To scrutinize the proposed framework, we analyze the energy consumed per MapReduce job, both in the data center and P2P models. Here, we characterize the Hadoop implementation of MapReduce, as depicted in Figure 4.2 with five phases of Map, collect, split, combine, shuffle and Reduce. Map, collect, split, and Reduce are generally accomplished in hosts, while combine and shuffle are network and storage hungry phases.

4.3.1 MapReduce Data Flow Analysis

Recalling from previous chapter, here we analyze what happens when a MapReduce request is sent to a data center. The input is split into n_t inputs of S_t in the map phase, to assign the tasks to the hosts. Each individual task with specified input is allocated to a host in the data center. To complete a task, a host acquires not only the task input data, but also the appropriate VM containing the execution code. Therefore, the data transmitted within the data center communication infrastructure includes VM and input data. For input data size, S_{in}, we assume several cases from the range of 1GB to 50GB.

In the Reduce phase, output is aggregated in the output file of size S_{out} and delivered as the job result. Besides, the Map phase output, which is called intermediate output, is exchanged among hosts as a result of the shuffle-exchange phase, with size S_{inter}. Therefore, the data to be transmitted is as depicted in Figure 4.2.

S_{VM} and $|hosts|$ denote the VM size and the number of hosts assigned to the job respectively.

The output data size and intermediate output size may vary according to the MapReduce application type and the input file. A diverse range of output and intermediate output sizes is elaborated across our evaluation scenarios, in this case study; however, in typical cases, we consider the intermediate output and output size to be 30% and 10% of input data size.

4.3.1.0.1 Data center communication: The energy consumed to transmit the required data for a job, as shown in (3.8), is the multiplication of power drawn for the communication by the amount of data that should be transmitted, as depicted in Figure 4.2, over the network throughput, τ_{DC}. Considering three-tier, switch centric network architecture for intra-data center communication, recalling from (3.8), if we assume a one hop path to traverse in each tier, the energy consumption in this part follows (4.1). $P_{switch}^{intra\_DC}(l)$ represents the power drawn in the switch at layer l, in the three-tier architecture. Note that we assume homogeneous replication factor and strategy in all the levels of the data center network. S_d represents the size of the data that should be transmitted over the data center neworks, which can be computed according to the formula in Figure 4.2. $P_{switch}^{intra\_DC}(l)$ denotes the power drown in layer l of the intra-data center communication.

$$E_{intra\_DC\_comm}^{MR} = \frac{S_d \times r}{\tau_{DC}} \times \sum_{l=1}^{3} P_{switch}^{intra\_DC}(l) \tag{4.1}$$

4.3.1.0.2 P2P communication: To analyze the energy consumed in the P2P-cloud per MapReduce job, we should consider two different scenarios. A case where jobs are assigned to the hosts within a vicinity, i.e. intra-vicinity, and the second case for inter-vicinity responses. In case of inter-vicinity responses, a job may be assigned to hosts in another vicinity. The input data, intermediate output and VM images should be sent to the distant host through the Internet. On the other hand, in case of intra-vicinity responses, VM, input and intermediate output data

need only to be sent to a host via wireless network. To exemplify, considering IEEE 802.11a wireless infrastructure and TCP protocol, we obtained throughput $\tau_{intra\_P2P} = 10.9Mbps$, with the average number of hops to traverse of 2 for the access points and 4.78 for routers, as we discussed in the previous chapter. Overall, the energy required to accomplish a MapReduce job on community for the intra-vicinity mode is given in (4.2).

$$E^{MR}_{intra\_P2P} = \frac{S_d}{\tau_{intra\_P2P}}(2P_{AP} + 4.78P_{router}) \tag{4.2}$$

In inter-vicinity mode, besides the energy consumed in the vicinity, some energy is dissipated for transmitting the input data and VM images to the far vicinities. Therefore, the energy drawn in such scenarios have an added term to (4.2), and is explained as $E^{MR}_{inter\_P2P} = E^{MR}_{intra\_P2P} + S_{in} + (n_{hosts} \times S_{VM}) \times E_{Internet}$

$$E^{P2P}_{c} = \sum_{h \in hops} P_c(h) \times \frac{r_h \times S_d}{\tau_h} \tag{4.3}$$

4.3.1.0.3 Communication over the Internet: Further to intra-data center or intra-vicinity communication in P2P, input and output data should be transmitted over the Internet, in the data center and inter-vicinity service provisioning cases of P2P service provisioning. Recalling the Internet energy consumption as modeled in the previous chapter, the energy consumed over the Internet follows:

$$E^{MR}_{Internet} = \frac{S_{in} + S_{out}}{\varphi} \times \sum_{* \in L} \frac{P_c(*) \times |hops(*)|}{\tau_*} \tag{4.4}$$

4.3.1.0.4 Computing Energy: The energy drained within each host is $P_{host} \times t_{task}$ for each phase. t_{task} is the time to process the assigned task in the host. Therefore, the overall computing energy dissipation follows (4.5).

$$E^{MR}_{compute} = \sum_{i \in phases} \sum_{j \in n_t} P_{host}(i,j) \times t_{task}(i,j) \times r_{ij} \tag{4.5}$$

TABLE 4.1: VM Specifications for MapReduce Case Study

Type	Cores	Memory (GB)	Storage (GB)	number of mappers	number of reducers
Small	1	1	1	1	1
Medium	1	3.75	4	1	1
Large1	2	7.5	32	1	1
Large2	2	7.5	32	2	2

4.3.1.0.5 Caching: As stated, the MapReduce workload is composed of input data, intermediate output data and VMs that contain the computing platform. A remarkable amount of energy is consumed to transmit the VM packets over community network. To alleviate the burden of VM transmission, in this work, we introduce the caching mechanism to save most prevalent VMs in community nodes, i.e. P2P-cloud with cache. In this way, we can save the energy required to transmit the VMs over the community each time.

4.3.2 Experiment Setup and Scenarios

In this section, we study MapReduce service provisioning in clouds with more details. We analyze the energy consumption on P2P-cloud and data center models under the MapReduce workload with the following configuration.

We follow the same hardware configuration as stated in Chapter 3. Recalling from the previous chapter, for the P2P-cloud nodes we rely on the Clommunity [1] which employs the Jetway JBC362F36W with Intel Atom N2600 CPU with the maximum power of 20W, as well as the Dell OPtiplex 7010 desktop machines. Data center hosts are set to be HP Pro Liant servers equipped with Intel Sandybridge CPUs. The community cloud infrastructure is modeled as wireless network which employs wireless antenna consuming the maximum of 5.5 watts. For the data center switches , we apply the power values computed according to the data provided in the switches data sheets; Internet energy consumption values are derived from [10]. Four VM types as shown in Table 4.4 are employed.

[1] `http://wiki.clommunity-project.eu/`

For most scenarios, we assumed a typical workload of input data size of 15 GB, overall intermediate output size is 30% and the final output size is 10% of the original input. For the sake of comparison through this evaluation, we take small VMs to execute the tasks, unless it is explicitly mentioned. We study energy consumption in the following scenarios:

A. **P2P-cloud without cache:** the base P2P-cloud scenario, assuming that the entire contents of workloads have to be downloaded via wireless, but are always available within the vicinity.

B. **P2P-cloud with cache:** same as above scenario, but enhanced with caching locally to the nodes the most popular VMs and data files within the vicinity, thus reducing the amount of repeatedly downloaded information. Note that in this scenario and the scenario above we assume that resource scarcity never occurs.

C. **P2P-cloud with inter-vicinity responses:** the worst case P2P-cloud scenario, the base one but the content is not available within vicinities, thus accounting for inter-vicinity communication and extra costs.

D. **P2P-cloud with cache and inter-vicinity responses:** same as above, extended with local caching of VMs and data files, thus reducing the amount of repeatedly downloaded information.

E. **Classic data center:** For comparison against the classic data center scenario, where users access the data center exclusively through wired networks, we exploit the data center model with 4 rows of 32 clusters each with 32 hosts for the data center model.

4.3.3 P2P-cloud Energy Consumption

In Figure 4.3, we show the energy consumption for each of the defined scenarios as the workloads vary across two parameters, VM size and input data size. Naturally, energy consumption increases for workloads executing larger VMs and

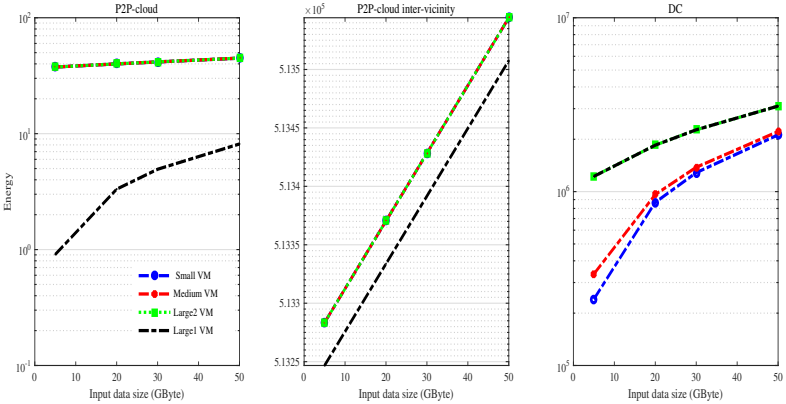


FIGURE 4.3: Energy consumption for various inputs across scenarios

when processing larger input data files. Comparing to the classic cloud, P2P-cloud consumes quite less energy as long as the jobs are performed locally. Generally, the energy required to accomplish jobs in data center model exceeds that of the P2P-cloud in any cases if the input size is big enough or the VM is large. However, we should bare in mind, this energy saving occurs by sacrificing the performance.

As shown in Figure 4.3, the energy consumption in P2P-cloud in case of providing the service within the vicinity is much less than the case of inter-vicinity scenario, since in the inter-vicinity service provisioning we should transmit the input, output data and in some cases the VMs through Internet, which is the most energy hungry element of the P2P-cloud system. In general, the communication energy is fluctuating more in P2P-clouds, while the processing energy is more varying in classic data centers.

4.3.4 VM size effect

As shown in Table 4.4, we consider three different types of VMs with different capabilities of processing MapReduce tasks. Figure 4.3 highlights the effect of

VM size in MapReduce task processing in three scenarios. As depicted, the energy consumption in P2P-cloud intra-vicinity processing is neutral to VM size, but is dependent of the MapReduce task processing slots available in the VM. By including more slots in a VM, we save more energy, since less communication overhead is induced. The energy consumption of communication in P2P-cloud constitutes an enormous portion of the consumption and even more than computation cost. Although increasing the level of parallelism within a VM can improve the energy saving, it should be bared in mind that in P2P-cloud the processing power of the nodes is very limited and we cannot develop large VMs there. Nevertheless, increasing the task co-location in classic data center hosts can be a more practical solution for energy saving purposes. As shown, energy consumption of inter-vicinity scenario is independent of the VM size as long as the VM images are available in the serving vicinities, since the input and output data transmission energy dominates the process energy consumption.

Increasingly, Figure 4.3 reveals the importance of choosing the right VM according to the input size besides choosing the appropriate platform. To exemplify, in a classic data center for the input size of less than 10 GByte, processing on small VMs is the most energy efficient choice due to the process power saving of small VMs.

4.3.5 Input-(intermediate) output Proportionality

Here, we study the relation of intermediate output and output size of the MapReduce applications on the energy consumption to get an insight into the appropriate VM as well as system to run different MapReduce applications. Figure 4.4 illustrates the importance of VM selection for applications with smaller input and output sizes. As shown in Figure 4.4, in cases that input size is small, i.e. 5GB and the output is less than 40% of the input data, data center model outperforms the inter-vicinity scenario.

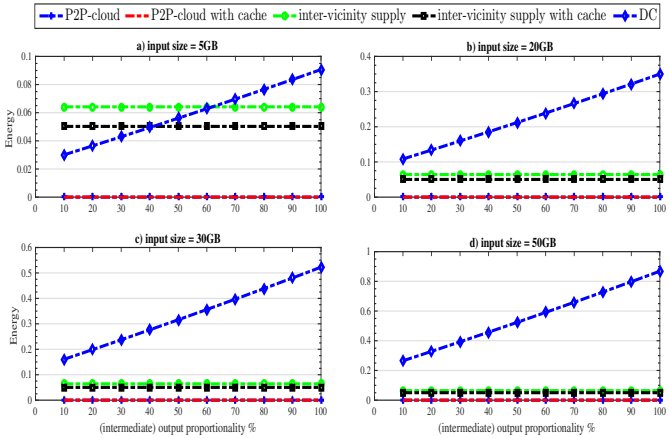


FIGURE 4.4: Energy consumption of applications with different input-output sizes running on small VMs

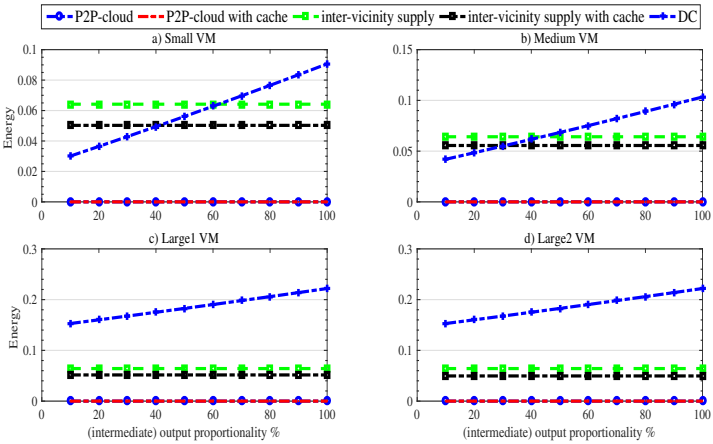


FIGURE 4.5: Energy consumption of a 5GB input application running on different VMs across scenarios

Figure 4.4 focuses on small VMs. To be more precise, we draw the energy consumption for small inputs across different scenarios including different VMs in Figure 4.5 because the intermeidate-output has to be exchanged among vicinities in this case. As depicted in Figure 4.5, in small and medium VMs there is a cross point among data center energy consumption and inter-vicinity responding

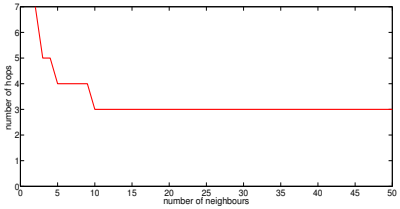


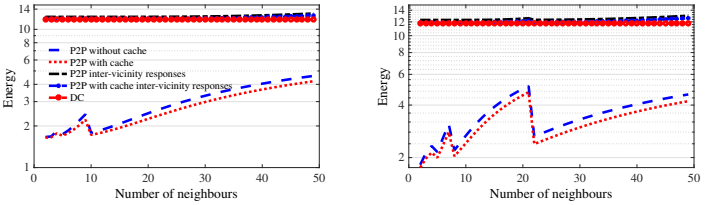
FIGURE 4.6: Impact of number of neighbours in vicinity diameter on average hops between two nodes.

in P2P-cloud, Figure 4.5.a and Figure 4.5.b. However, for the large VMs, energy consumption of data center always exceeds the P2P-cloud scenarios even for the small input size, Figure 4.5.c and Figure 4.5.d.

4.3.6 Vicinity Density

Here, we exert the logarithmic vicinity diameter model which implies the average distance of two nodes in the vicinity as $O(log^{n}_{neighbourCount})$ where n denotes the scale of the system. Figure 4.6 shows that with the number of neighbors of at least 10, P2P-cloud scenarios can keep the average number of hops between two nodes in the vicinity, where there are 100 nodes overall in the vicinity. Convergence to three hops for a vicinity of 500 nodes occurs in around 30 neighbours. Although three hops is very effective, increasing the number of neighbours not only leads to higher energy consumption due to multiple unaddressed recipients, but also does not provide additional gains in message latency. Nonetheless, adding more nodes increases the resource availability in each vicinity. Therefore, there is a trade-off between energy efficiency and resource availability.

In Figure 4.7, we depict energy consumption for the typical workload presented earlier for all the scenarios described, with two different vicinity sizes: 100 and 500. P2P-cloud with caching, our proposal, is clearly the winner, with orders of magnitude less energy consumed, in both scenarios. Figure 4.7 also reveals the influence of the vicinity density, i.e., the number of neighbors accessible to each node. The P2P-cloud with caching is always the winner regardless of the vicinity



a) Vicinity of 100 Nodes b) Vicinity of 500 nodes

FIGURE 4.7: Vicinity Density effect in community networks

density. The fluctuation in the graph for small number of neighbors is because of the estimation and round up error in the logarithmic vicinity diameter model, but by reaching the efficient average hop count, i.e. three for aforementioned scenarios, energy consumption rises gradually as the vicinity becomes denser.

4.4 Evaluation

In this section, we study different cloud scenarios to highlight the pros and cons of different systems. We aim to analyze the energy consumption on different cloud models under the video streaming, storage and MapReduce workloads. We conclude this section with the evaluation of proposed framework.

All the results in this chapter are derived from the same hardware setting explained in the previous chapter, summarized in Table 4.2. In this section, we compare data centers, with different granularity as explained in Table 4.3, with P2P-cloud which includes 20% of Dell machines in the testbed and the rest are Jetway devices. Three VM types, as shown in Table 4.4, are exerted in our study.

TABLE 4.2: Summary of experiment network setting

		dynamic power (one port)	**static power**
DC	core	31.25W	100W
	distribution	9.56W	80W
	commodity	1.56W	0W
P2P	TP	1.3W	3.7W
	NS	1.1W	3.9W
Internet	Core	0.15W	–
	Metro	10.25W	–

TABLE 4.3: Studied data centers

Type	hosts/cluster	clusters	data centers	hosts/DC	PUE
Nano	1	1	2^{20}	1	1.000
Medium	32	32	1024	1024	1.032
Large	32	1024	32	2^{15}	1.110
Mega	32	32768	1	2^{20}	1.330

TABLE 4.4: VM Specifications

Type	Cores	Memory (GB)	Storage (GB)
Small	1	1	1
Medium	1	3.75	4
Large	2	7.5	32

4.4.1 Idle case energy consumption

As stated, idle power consumption is one of the obstacles that impedes the attainment of energy proportional systems. Here, we study the idle case power of different data center models with the same processing capability, and the corresponding network infrastructure considering hierarchical network topology. In this situation, to be able to compare different cases, we should follow hierarchical network topology, because the switches that we introduce for our evaluation, can only support 2^{13} hosts in a data center of Fat-tree topology, which is not scalable enough for the mega data center case.

Nonetheless, conforming to the server-centric topology of CamCube, we observe that the network idle power is negligible due to the green server ports that consume almost zero watts in the idle case. Moreover, CamCube constitutes a network of servers via peer-to-peer connection, which is exempt from any network hardware switches.

As illustrated in Figure 4.8, power is significantly drawn in the hosts of the cloud, except for the P2P-cloud which consumes more power in the communication aspect. As explained in the previous section, the P2P-cloud hosts must connect to an Access Point (AP) which is connected to a Nano Station (NS), and each of these devices draws 4W in the idle case. Moreover, this figure highlights the effect of data center granularity on idle power dissipation. The larger the data center is, the more power is drawn in the idle case. Therefore, moving toward distributed data

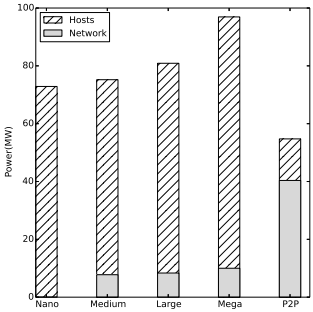


FIGURE 4.8: Idle power consumption across scenarios

center models with smaller sizes contributes to reducing idle power and promoting the overall energy proportionality of the cloud.

4.4.2 Service energy

Figure 4.9-a and 4.10-a show the energy consumption from the service perspective in streaming and MapReduce applications. Results in Figure 4.9-a and 4.10-a indicate that the energy required for processing in cloud systems is much higher than the energy required for transmission, except for the offline video streaming in P2P-clouds.

Note that as shown in Figure 4.9-a, in offline streaming, the P2P-cloud consumes the most energy in the transmission phase, while the energy dissipated for transmission in different data center models is almost the same for both online and offline streaming.

However, processing energy is increasing gradually in line with data center granularity for online streaming. The processing draws the same energy in data centers regardless of the data center size, because it includes retrieving and processing the same amount of data from the storage. Note that throughout this section, we have assumed that each user and all data center hosts have the same video streaming VM already installed on their machines. This is not an unrealistic assumption as its cost would be quickly amortized after a small number of videos served and

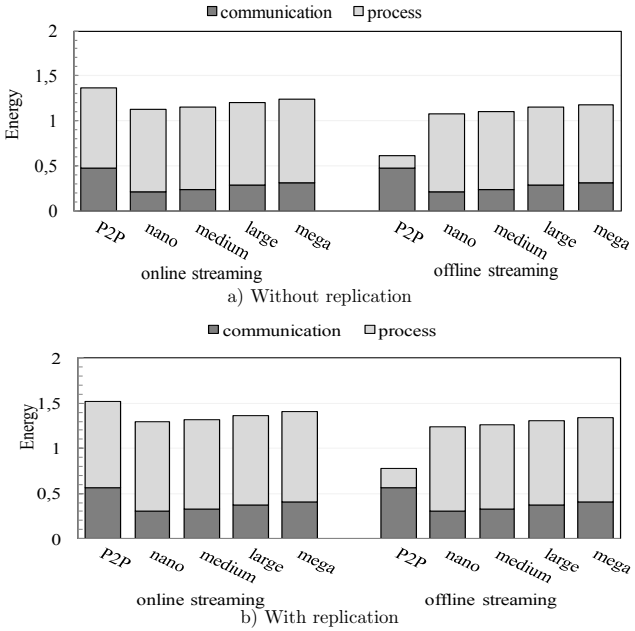


FIGURE 4.9: Video Streaming

played back. Nonetheless, the VM size can affect the energy consumption in the processing.

Figure 4.10-a illustrates that for MapReduce WordCount applications, the transmission energy consumed in P2P-cloud is noticeably higher than that consumed in the data center model, because nodes are not connected through an energy-efficient network as in the data center. Moreover, we see that the nano data center model dissipates slightly more energy for transmission due to additional inter-data center communication required to interact with different elements of the system in each MapReduce phase. Note that, we assume all the MapReduce tasks are accomplished within a single data center in the distributed data center model. This is a favorable, yet reasonable scenario. Nonetheless, transmission energy can be remarkably higher if we have to share the tasks among the data centers. Conversely, the processing energy consumed in the P2P-cloud is slightly lower than

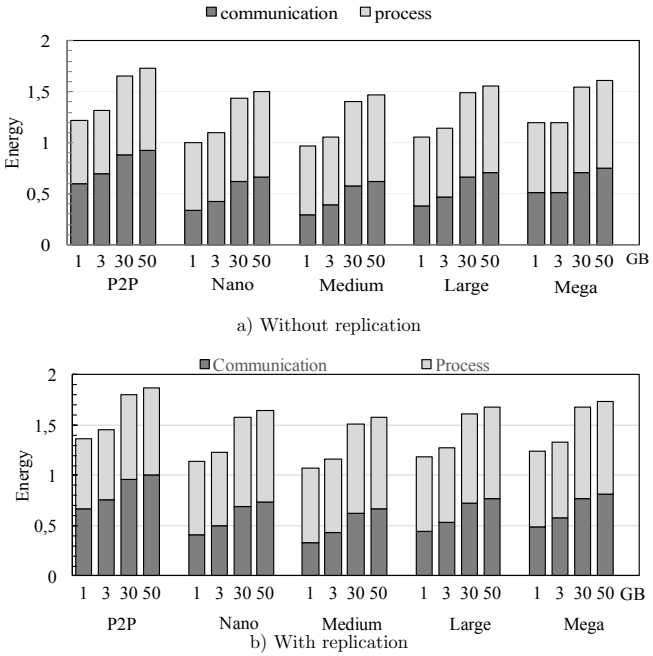
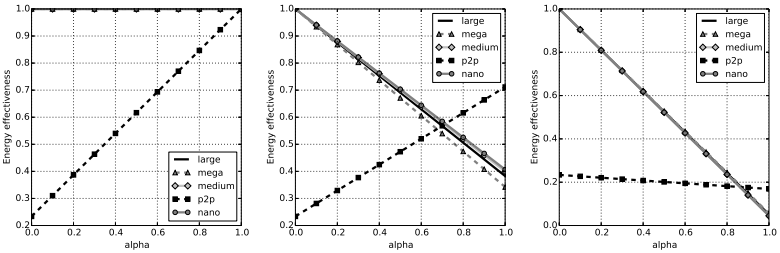


FIGURE 4.10: MapReduce WordCount Application

in the data center model due to more energy proportional hosts and no need for cooling mechanism. Data center processing energy is increasing in line with the granularity as a result of greater cooling energy required.

4.4.3 Replication

Replication of offline video streaming literally increases the number of tasks and data size times to the replication factor. Therefore, replication increases the probability of finding locally available content and increases the reliability of P2P-cloud service provisioning. Figure 4.9-b and Figure 4.10-b illustrate the energy consumption in the presence of replication for streaming and MapReduce applications, respectively.



a) Offline video streaming b) Online streaming c) MapReduce applications

FIGURE 4.11: Energy Effectiveness across scenarios

In video streaming scenarios, we consider the replication factor of 3 for the content distribution. Comparing Figure 4.9-a and Figure 4.9-b, we find out that offline video streaming over the P2P-cloud is more energy efficient, even with replication, in comparison to the data center model, provided that we have the content stored locally in the vicinity.

In the case of MapReduce, common Distributed File Systems (DFS), such as Hadoop HDFS which is prevalently exploited, has a replication factor of 3, to replicate the input and output data. All the same, in the P2P-cloud, we have to replicate the tasks to insure the reliability, fault-tolerance, and performance. Due to higher dynamicity of P2P systems, compared to data center, higher replication factor is required, e.g. Tahoe-LAFS's default replication factor is set to 10. Therefore, replication affects the energy consumption in P2P-cloud more than in the data center model. In data centers, replication substantially affects the communication directly proportional to the replication factor.
-0pt

4.4.4 Energy Effectiveness

To compare the energy effectiveness of different data centers, as introduced in the previous chapter, throughout the studied services, we assume that running the application on the data center can provide the maximum performance. Therefore $\frac{\hat{\mathcal{P}}}{\mathcal{P}^*}$ for data center service provisioning is equal to one, while this value is in the

range of zero to one for the P2P-cloud. However, P2P hosts consume less energy to process the applications, which leads to values higher than one for the term $\frac{E^*}{\hat{E}}$ in (3.5). As a result, in Figure 4.11, we can find the energy effectiveness of P2P-cloud to be higher than of the data center by orders of magnitude when α is large enough ($\alpha > 0.7$).

Nonetheless, as the energy weight decreases, energy effectiveness decreases in the P2P-cloud. This partially indicates the correlation of energy proportionality and energy effectiveness. The heterogeneous environment of the P2P-cloud delivers a more energy proportional processing environment compared to the data center with homogeneous servers; we quantify the heterogeneity effect, later in this section. Therefore, the higher the processing power, the higher the chance of promoting the energy effectiveness of the P2P-cloud via increasing the α. Hence, the energy effectiveness of the P2P-cloud offline streaming is always lower than that of the data center model, as shown in Figure 4.11-a. This happens because little processing is conducted in the cloud side to retrieve the video from storage, and the transmission energy is higher in the P2P-cloud model. For MapReduce applications, Figure 4.11-c, the data center model is more energy effective unless the system is very energy conservative via setting $\alpha \geq 0.9$.

Figure 4.11-b,c highlight that the performance of the online streaming and MapReduce application, regardless of the data center granularity, does not change, exploiting the same infrastructure hardware ($\alpha = 0.0$) provided that we have sufficient resources in smaller data centers. However, increasing the α value causes the mega data center's energy effectiveness to decrease more, due to higher PUE.

Vulnerability Factor: Figure 4.12, indicates the vulnerability factor for each application to the α parameter, in the context of different platforms. From the figure, we can see that the P2P-cloud's energy effectiveness varies in the positive range while the data center platforms show variation in the negative range. Positive vulnerability in the P2P-cloud is associated to the inefficient hardware in terms or energy, which makes energy term more vulnerable to α tuning.

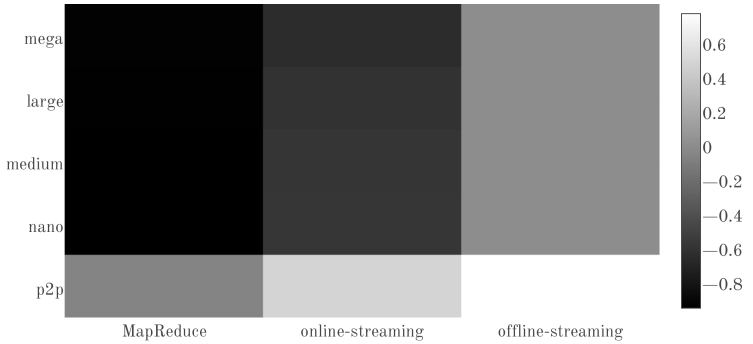


FIGURE 4.12: Sensitivity to α selection across different services and platforms

On the other hand, the negative vulnerability values in data center platforms indicate that in these platforms the performance term plays the dominant role in the final energy effectiveness value. Moreover, from an application perspective, the least computing intensive one, i.e. offline-streaming is less vulnerable to α miss-configuration in data center model. However, in P2P platform, it is significantly influenced by the P2P system dynamics, e.g. connection instability.

Relative Energy Effectiveness: As shown in Figure 4.13, the relative energy effectiveness ($\mathfrak{E}^{\mathbb{R}}$) of offline streaming, in the data centers with different granularity, is around 0.7. All the same, the online streaming keeps close $\mathfrak{E}^{\mathbb{R}}$ value for data centers, regardless of their granularity. The little differences are due to the various levels of energy conservativeness in the studied data center models.

However, this value varies for all the services in the P2P-cloud platform due to the diversity of hardware which leads to different power dissipation and fluctuation in performance because of system dynamics.

We see in the figure that P2P-cloud shows better relative energy effectiveness for offline streaming, due to the local service provisioning, which cuts down the communication energy dissipation remarkably in this communication intensive application. Nonetheless, data center models perform better in the other applications which are computing intensive. The more computing intensive the application, the bigger the gap between the data center and P2P-cloud models.

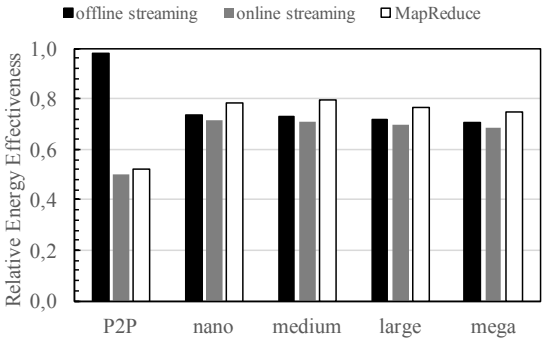


FIGURE 4.13: Relative Energy Effectiveness of different services across the scenarios

4.4.5 Communication pattern

Here, we elaborate on the energy consumption to transmit 1GB of data on different topologies in three scenarios: i) intra-cluster scenario which targets the communication among the servers of the same cluster. ii) inter-cluster communication that focuses on the transmission of data among the servers of two different clusters. iii) inter-data centers that looks at the communication among different data centers of a distributed cloud.

Figure 4.14 reveals that the CamCube server-centric topology consumes the most amount of energy in intra-cluster and inter-cluster scenarios. Among switch-centric scenarios, Fat-tree dissipates the least energy in the intra-cluster scenario. Flattened butterfly is the most energy efficient for inter-cluster communication, but it is not easily scalable for the case of the mega data center. For the inter-dc scenario, the difference among the topologies decreases, because the inter-dc connection energy consumption dominates the other components energy consumption. As a result, a hybrid topology that combines the best of all components of the data center network can be a solution for a more energy efficient topology.

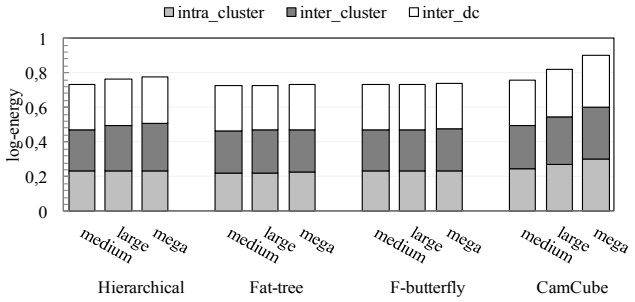


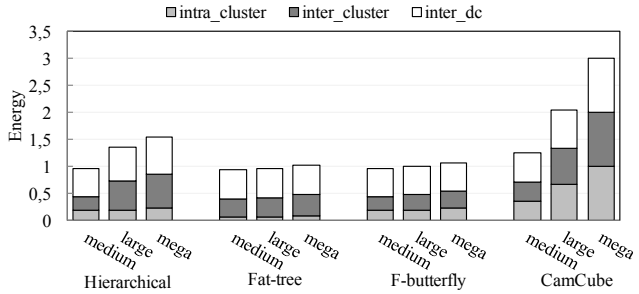
FIGURE 4.14: Topology effect

4.4.6 VM Migration

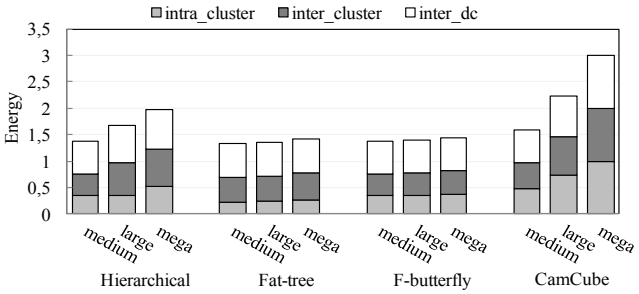
Because processing is more a function of VM activity log, and can be considered the same for the identical application with the same input, we only consider the communication energy for VM migration. Across Figure 4.15, the energy consumption due to VM migration is elaborated for different VMs mentioned in Table 4.4. As we see in the Figure 4.15, migration energy dissipation conforms to the communication pattern energy consumption directly proportional to the data center granularity and networking, as studied in Section 4.4.5. All in all, the VM migration within a cluster is the most efficient approach in terms of energy, regardless of the VM size.

4.4.7 Energy Proportionality in Heterogeneous Environment

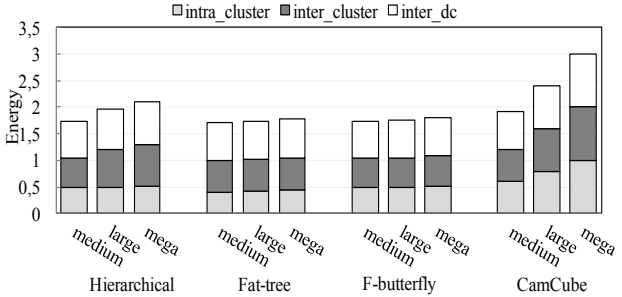
As explained earlier, state-of-the-art machines draw power in an energy non-proportional pattern, primarily due to the high idle power consumption that dissipates the most power in cloud systems, as shown earlier in this section. Some investigations focus on energy-proportional hardware design. Nevertheless, in the architecture level, by combining the heterogeneous processing elements, we can improve the energy proportionality of the system and even surpass the ideal case



a) Small VM



b) Medium VM



c) Large VM

FIGURE 4.15: VM migration across scenarios

boundaries in specific situations [37]. Typically in data centers, servers are homogeneous machines to alleviate the burden of management issues. However, the scheduling mechanism can be smarter, if it is equipped with the load level of each node and can decide based on energy proportionality [13].

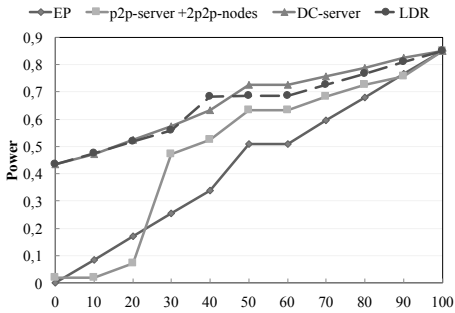


FIGURE 4.16: Heterogeneity effect on energy proportionality

Here, we elaborate on the scenarios in which the processing capacity of each combination of hosts is equal to a data center server machine. In Figure 4.16, we see that the combination of P2P-cloud server and host nodes consumes even lesser power than the energy proportional server. Moreover, this figure implies that combining the data center servers with P2P nodes contributes to improving overall energy proportionality of the hosts, and at low utilization (i.e. less than 20%), we can cross the proportional server boundaries in terms of elevating the hosts energy proportionality. In the range of 40% - 60%, we see the improved proportionality of P2P-cloud nodes compared to heterogeneous data center hosts.

4.4.8 Resource Availability vs. Static Power Dissipation in P2P-cloud

Figure 4.17 shows the pattern of static power dissipation by increasing the number of nodes and super nodes proportion. This figure illustrates that increasing the computing resource availability may lead to more static power consumption. Therefore, increasing the resources in P2P is not always the best solution. However, an out of the box solution can be adding cache to each vicinity to decrease the VM image communication energy, as depicted in Figure 4.18. As we see in this figure, for MapReduce applications, where we need to install a lot of VM images in the hosts, applying caching techniques significantly contributes to the energy

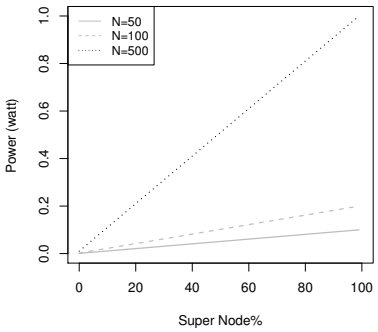


FIGURE 4.17: Resource availability vs. static power consumption

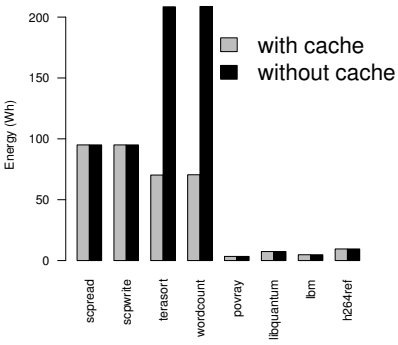


FIGURE 4.18: Effect of using cache in P2P

saving. Nonetheless, the saved energy is not remarkable for the other applications due to their single thread nature, and comparatively small code.

4.4.9 P2P-assisted Cloud Efficiency

Here, we study the prospective energy savings by applying our proposed resource allocation mechanism in P2P assisted cloud ecosystems, under the experiment scenarios explained in the previous chapter. Note that we chose these applications due to the more pragmatic view they provide. Figure 4.19 shows the energy that can be saved by serving the applications in the proposed framework. As illustrated, a remarkable proportion of energy can be saved in file transfer and MapReduce applications, while, for process intensive applications of scientific computing and image processing this portion is limited, due to the process intensive nature of them.

4.4.10 Put It All Together

Overall, referring to the results provided in this section, we can see that for the services requiring distributed data processing and, hence, communication among

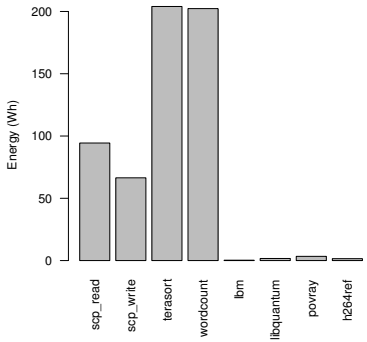


FIGURE 4.19: Energy saved per application in the framework

the processing elements, data center model energy effectiveness outperforms P2P-cloud, while for the content delivery based services (e.g. offline video streaming) P2P-cloud is the predominant model in terms of energy saving. However, considering the performance by defining energy effectiveness, the table may turn in favor of the data center model due to limited resources and processing capabilities of the P2P-cloud. Therefore, a data center assisted P2P-cloud model can fulfill the energy effectiveness ambitions. There are efforts on providing the streaming services based on such an architecture, namely CLive [100] which adaptively leverages cloud based resources for P2P streaming of video. Nonetheless, P2P-cloud architecture can affect the energy efficiency. Thus, forming an energy efficient architecture can improve the service energy effectiveness as well.

Loosely paraphrasing, a bi-level, hybrid, distributed cloud architecture which combines the concept of P2P-cloud at the edge of the system assisted by local data centers at higher level is an architecture-level solution to improve the energy effectiveness of cloud services. Bi-level architecture is the most energy effective method. However, it warrants the complexity of interoperability, mobile agents, security, resource discovery, and management.

What is more, at the data center level, granularity affects the energy consumption. Smaller data centers are more flexible and induce less PUE, considering the same

technology. Moreover, they are exempted from dealing with scalability issues in network topology as in mega data centers. They can easily adapt to any network topology to improve the performance and energy consumption. Furthermore, a distributed data center architecture brings high availability which contributes to local service provisioning and provides further opportunities to access local renewable energy sources.

The above results witness that no global answer exists for the energy conservativeness of services in the comparison of P2P-cloud and data centers. This is an evidence to prove the vital importance of the introduced framework.

4.5 Summary

In this chapter, we introduced a hardware agnostic framework to analyze energy effectiveness both from hardware and service vantage point. Characterizing the energy model for a particular setting and leveraging this framework, a broker can decide for a more energy effective service allocation, which considers energy conservativeness while still meeting the quality of service requirements. Moreover, we compared the energy consumption in classic data center model with P2P-clouds. We scrutinized the main sources of energy consumption in both systems and assessed their energy effectiveness. Employing this framework, we can compare the energy effectiveness of each individual service in given hardware settings of P2P-cloud and data centers and opt the most energy efficient platform accordingly.

We presented the P2P-cloud in intra-vicinity scenario, as a response to a more energy efficient solution to assist cloud ecosystems. The effect of exerting P2P-cloud on quality of service is studied on services for file transfer, video streaming, and MapReduce jobs. Our MapReduce case study indicates that the hardware specifications may completely turn the table in favor of a specific platform which revokes the possibility of finding a straight forward answer to our major question in this research, i.e. "Is it energy efficient to switch to community cloud?", and highlights the necessity of a general and adaptive framework to work as a

middleware between the service stack layers. This middleware framework maps the hardware-agnostic, coarse-grained service level model to a particular hardware setting. This mapping refines the model and comes up with an answer for this question, "Which platform is more energy efficient for the considered service?"

Part III

Combining P2P-assisted Cloud with Smart Grid

—"The limits of the possible can only be defined by going beyond them into the impossible."

-Arthur C. Clarke

5

Combining P2P Assisted Cloud and Smart Grid

As carbon footprint rate rises in recent years, and is predicted that the global carbon emission will reach 1430 megatones by 2020 [1], being energy efficient and moving toward green energy sources are essential for environmental sustainability. Information and Communication Technology (ICT) plays a leading role in this context by its potential for providing a large scale real time controller to improve decision making and developing environmental information systems [88]. Moreover, investments in energy saving technologies are compensated financially, particularly, when carbon tax is applied to the energy price. However, this ICT infrastructure itself is a source of energy consumption. For instance, cloud computing energy consumption will increase to 1,963 billion kWh by 2020 and the

associated CO2 equivalent emissions of 1,034 megatones will be expected [1].

Therefore, green ICT and ICT for green are not mutually exclusive, both are important and they complement each other [40]. Hence, the challenge for the future lies in synthesising, not only ICT for green, but also green ICT, to achieve a more sustainable service platform.

The electricity industry is transforming from a centralized, producer controlled network to a more decentralized and consumer interactive one via smart grid. Smart grid intends to achieve grid's full potential and prepares a cleaner and more efficient, reliable, resilient and responsive electric system. A smart grid system needs a large scale infrastructure for collecting and communicating data. Likewise, it must have access to flexible, network-scattered computational power, network bandwidth, and storage capacity, due to distributed nature of data sources.

Akin to the smart grid, ubiquitous P2P society is a collaborative effort in which infrastructure and services are shared among several individuals and/or organizations forming a specific community with common concerns. Ubiquitous society envisions a world in which services are accessible from anywhere, anytime, by anyone and anything[1]. These goals are partially intersected with the cloud vision, which introduces pervasive service provisioning. Therefore, we name the ubiquitous P2P society as P2P-cloud.

Since energy and ICT are two pillars of modern life that advance hand in hand, in line with the goals of ubiquitous society, in this chapter, we propose Cloud of Energy(CoE) system, which considers everything as a service (XaaS), as introduced in the idea of clouds [80], e.g. Infrastructure as a Service, Platform as a Service, Software as a Service. In tandem with this trend, Energy as a Service is added to the agenda in CoE. Smart grid and P2P-cloud are both large scale distributed systems involving numerous common specifications: self service, metered, elastic resources, multi-tenant, and access via the network are cases in point. Thus, CoE combines P2P-cloud, including sensors, commodity desktop machines

[1] `http://www.itu.int/WORLD2006/forum/ubiquitous_network_society.html`

and IoT boards, with the smart grid, to provide energy efficient services and also contributes to the smart energy system's computing and communication platform.

There is a growing body of work centered on exploiting the cloud and peer to peer platforms for the smart grid computing [42, 92, 111, 123]. In a cloud computing environment, flexible data centers offer scalable computing, storage and network resources to any Internet-enabled device on demand. Moreover, P2P-cloud can manage the massive amount of data from distributed sources of consumption, generation and network nodes. On the other hand, diverse energy sources of smart grid improves the availability, sustainability and environment friendliness of the ubiquitous network society services.

The major contribution of this part is providing a qualitative comparison of P2P-cloud and smart grid in Section 5.1, and accordingly introducing the CoE architecture as an integrated energy and computing platform, Section 5.2. CoE aims to design a service framework that incentivize all range of service producers, offering services from computing to energy, in range of small prosumers to giant providers, to serve in a greener marketplace, through an economic middleware. We analyze the feasibility of the proposed architecture in Section 5.5. Relevant publications to the contributions in this chapter are [114–116].

5.1 Background and Related Work

The electricity industry attempts to transform itself from a centralized, producer controlled network to a more consumer interactive and decentralized one via smart grid. Akin to the smart grid, P2P-cloud is a collaborative effort in which infrastructure and services are shared among several organizations form a specific community with common concerns.

The economic models invented for smart grid and P2P-cloud systems are inspiring for each other, on account of the similarities of these two systems. Primarily, because in both P2P-cloud and smart grid, consumers can be providers as well.

Moreover, they both follow the pay-as-you-go mechanism and the computational tasks are not batched; hence, there is no waiting time. This enables a time-critical model of computation.

We can leverage the ad hoc and elastic nature of clouds to benefit the smart grid at the economy of scale expected from cloud computing, while efficiently utilizing power as we scale up. On the other hand, combining smart grid and community clouds in a symbiotic relationship can be mutually beneficial in fostering adoption of both.

Some previous work [27, 86, 92, 111, 123, 140] sketches a smart grid communication and information platform that relies on a cloud system. To employ such solutions, the smart grid should establish its own cloud system or must use public cloud infrastructure. In both cases, the smart grid should spend enormous amount of money for communication and data provisioning. Since the P2P-cloud is a community of the available end user resources, employing commodity hardware, no extra resource investment is necessary to manage a smart grid by exerting the P2P-cloud.

To this aim, we can integrate the pricing mechanism of both systems. The users supply the P2P-cloud resources for the smart grid, charge it according to their contribution and energy consumption and the community users have the opportunity to choose the platform with least energy cost to execute their services. This synergetic solution encourages the user to share as much resources as possible in the P2P-cloud, and eventuates to the end user utility expense reduction, as well. Furthermore, the P2P-cloud facilitates the energy efficiency issues by employing the efficient energy provisioning capabilities of the smart grid.

At the same time and increasingly so, the need to reduce carbon footprint has greatly raised investment in heterogeneous renewable sources of energy such as water, waves, wind and sun for an energy efficient smart grid. As stated in [46], the smart grid infrastructure is a combination of smart energy, information and

communication subsystems. Utilizing both information and communication systems, the smart grid accomplishes precise matching of supply to demand and offers incentives to appropriate consumer behavior. These changes affect the energy waste and the carbon footprint of the grid, making it smarter and greener.

Analogously, in the context of computing, replacing expensive, gigantic, cloud data centers by inexpensive P2P-cloud, constructed of commodity hardware, would be a huge step towards energy efficient systems. Previously, some studies, e.g. [73], compared the smart grid to the Internet. In the next section we survey the smart grid and P2P-cloud potential conjunction points.

5.1.1 Qualitative Comparison of Smart Micro-grid and P2P-cloud

As discussed so far, the design goals of the P2P-cloud appear to be nearly identical to those of the smart grid; the similarities and differences of smart grid and P2P-cloud are listed on Table 5.1. Both of them attain the basic requirements of a modern society in a large scale and distributed manner, namely electricity, communication and computing. Both infrastructures are conforming to a stochastic behavior due to resource fluctuation and highly evolving topology, regarding origin of requests and availability of resources. Loosely paraphrasing, due to the unpredictable collaboration paradigm of end users in the P2P-cloud, the system depicts a stochastic behavior.

Likewise, in the smart grid energy provisioning system, we observe a stochastic behavior of renewable energy resources participating in the system, which is tightly coupled with the weather condition of each geographical region. To exemplify, in a windy day, wind farms generate a lot of energy, while the solar panels reach their extreme productivity on a perfect sunny day. Both smart grid and P2P-cloud follow the bidirectional flow property, since most of the nodes collaborating in the distributed set of users and suppliers serve as prosumers, i.e. PROducers and conSUMERs concurrently, to respond to the distributed demand for energy

TABLE 5.1: Smart Grid and P2P-cloud similarities and differences

Features and Properties	**P2P-cloud**	**Smart Grid**
Scale	large	
Evolution rate	high	
Time Dependency	Time critical	
Billing mechanism	pay-as-you-go	
Resource behavior	Fluctuating resources	
Suppliers	Distributed	
Market behavior	non monopolistic	
Transparency	Consumers are unaware of the underlying complexity	
demand	Distributed and unpredictable	
Service Cost	Cost Effective	
Resource management	Distributed	
Hardware Costs	**Cheap**	**Expensive**
Range of services	**Diverse**	**Limited**
Storage Support	**Full support**	**some degree**
Availability	**Important**	**Critical**
(Byzantine) Fault Tolerance	Yes	
Scalability	Critical	
Reliability	Critical	
Consistency	Critical	
Data Security	Critical	

and information. Although the collaborative distributed systems supply the demands in a distributed manner, consumers are unaware of the underlying network complexity.

The most remarkable property of the both systems is "pay-as-you-go" mechanism employed in these systems, that eradicates the heavy investment for the centralized infrastructure. It revokes the supplier monopolies thanks to the decentralized, collaborative structure.

For both systems, the fundamental goal is to effectively integrate a number of separately administered existing networks into a common utility network. The

common secondary design goals are: 1) to tolerate loss of individual components, 2) to support different underlying infrastructure types, 3) to allow distributed resource management, 4) to be cost effective, 5) to allow easy endpoint attachment and 5) to be accountable for resource usage. To thrive a collaborative distributed network, we should consider a cost effective design as the most striking issue, while bearing in mind the fundamental design principles of a distributed system including scalability, reliability, availability, consistency, fault tolerance, distributed resource management and data security.

Moreover, some of the problems in the P2P-cloud, that of aggregation of stochastic sources, distributed resource management, multiple time scales of control, and user incentivization, are similar to that faced in the smart grid.

Nonetheless, it is not all about the similarities, there are some differences as well. The demand paradigm in the smart grid is more predictable than the P2P-cloud (more remarkable difference between peak and low usage); the electricity consumption pattern is almost fully determined beforehand in the smart grid. The peak demand time is almost predictable in the grid system, while it is not as easy to foresee the demand pattern in a distributed computing environment. Even so, many global services experience predictable peak and low periods for each time zone.

Furthermore, the diversity of provided services in the P2P-cloud is much higher than in the smart grid. This leads to the more complicated QoS and management mechanisms in the P2P-cloud. Additionally, the computing hardware costs follow a downward trend, while the hardware expenses of the smart grid rises day to day.

To design a comprehensive model for integration, we need to face the following challenges which stem from the natural differences of computing and energy systems.

- **Flow Management**: data flow management is way more flexible than energy flow management. In other words, we can encapsulate and label data

easily, while it is not easy to route the electrons in the same way. Thus, implementing Virtual Private Cloud (VPC) is easier than developing a Virtual Power Plant (VPP).

- **Storable Services**: in smart grid, batteries can save energy. Therefore, energy service can be stored instead of instantaneously offered to the demands, while it is not possible to store the computing service.

- **Stochastic behavior**: both systems are conforming to a stochastic behavior due to resource fluctuation and highly evolving topology, regarding origin of requests and availability of resources. In other words, due to the unpredictable collaboration paradigm of end users in the cloud, the system depicts a stochastic behavior. Likewise, in the smart grid energy provisioning system, we observe a stochastic behavior of renewable energy resources participating in the system, which is tightly coupled with the weather condition of each geographical region. However, the demand paradigm in the smart grid is more predictable than the cloud (more remarkable difference between peak and low usage). The electricity consumption pattern is almost fully determined in advance in the smart grid. The peak demand time is almost predictable in the grid system, while it is not as easy to foresee the demand pattern in a distributed computing environment.

- **Service Diversity**: the diversity of provided services in the computing platform is vaster than in the smart grid. This leads to a more complicated QoS and management mechanisms in the clouds.

5.1.2 Smart Grid and P2P-cloud collaboration potential

There is a growing body of work centered on exploiting the cloud and peer to peer platforms for the smart grid computing [92, 111, 123]. In a cloud computing environment, flexible data centers offer scalable computing, storage and network resources to any Internet-enabled device on demand. Moreover, P2P communication platforms can manage the massive amount of data from distributed sources of

consumption, generation and network nodes. On the other hand, diverse energy sources of smart grid improves the availability, sustainability and environment friendliness of the cloud services.

Smart grid aware ICT service provisioning can foster the idea of green ICT by better employment of energy sources. On the other hand, there are some endeavors to leverage ICT platform for smart grid communication and information subsystems. Besides, with the idea of Internet of Energy, Internet not only can serve as the communication infrastructure for the smart grid, but also the distributed mechanisms designed to manage the Internet and tackle the administration issues can inspire the solution space of smart grid challenges, which is called Internet thinking of smart grid [73]. All the same, the cloud is already proposed as the information subsystem for the smart grid, in the state of the art studies [42, 92, 111, 123]. Previous work suggests also how P2P-cloud can be leveraged as the information subsystem at the smart micro-grid level [114].

In [69], Niyato, et al. proposed a cooperative game based approach to manage the virtual machines of a cloud in a more energy efficient way by being aware of smart grid resources. In [92], analysing the power flow of data centers, authors formulated a service request routing mechanism that considers the load balancing of distributed data centers, which leads to energy consumption balance in data centers and helps to grid energy management.

Moreover, there are some studies[2] on how to leverage a Peer-to-Peer platform as the ICT infrastructure of smart grid. For instance, the CoSSMic project[2] aims to develop the ICT tools needed to facilitate the sharing of renewable energy within a neighbourhood. Cisco also proposed the combined platform of fog and cloud computing for smart grid data processing[29]. P2P clouds[45] and ClouT[127] approached this issue in a more general view by targeting the Internet of Things (IoT) enabled smart homes and cities.

A P2P-cloud-enabled smart grid can benefit from the following advantages:

[2] `http://www.cossmic.eu/`

- **Facilitating the development:** It is easier to develop P2P-cloud especially in the urban areas which facilitates the development of smart grid as well.
- **Providing communication and computing platform:** P2P-cloud provides both communication and computing platform while classic cloud relies on Internet for communication.
- **P2P-cloud offers user-enabled control mechanism:** A user can control the applications whereas they are open source or users can develop their own applications employing APIs such as REST.
- **Hierarchical data processing:** Smart grid data analysis on time series data perfectly matches the parallel data analysis. Data analysis algorithms can run on subsets of data, i.e. a subset of users' data chosen according to the locality property, stored on different machines, and aggregate them into the final result set through hierarchical, multi-level processing. As with the distributed storage, the distributed parallel processing is harnessing the network of commodity hardware to its fullest, in which the amount of available memory and computing power are abundant.
- **Contributing to the privacy preserving and service personalization:** Aggregation gives the possibility to anonymize data, which is a safe and secure way to retrieve business intelligence information to personalize the services without jeopardizing the end user privacy.

Nonetheless, smart grid can provide various energy sources for the community services. Charging according to the energy price, users are more concerned about the energy sources and prices, therefore, we make a broad range of choices for the users via providing the users with smart grid resource availability data. In the next section we introduce an architecture that facilitates the mutual collaboration of these systems.

5.2 Cloud of Energy

Partly inspired by Internet of Things (IoT), Internet of Energy (IoE) [33] is about providing energy as a service in a more efficient way by dynamically adjusting resources to deliver energy at the lower cost and the higher quality possible in the context of smart grid.

In line with the idea of Internet of Energy, here, we define Cloud of Energy (CoE). CoE outlines how involving customers in future ubiquitous society-driven energy conservation efforts can both foster the adoption of green energy, as well as green cloud due to the increasing energy awareness of society. The rationale is to get users into the loop, not only to guide them how to use the services, but also to involve them directly in the whole cycle of control, production and provisioning of energy. Ubiquitous society makes it possible to combine informational support with fostering intrinsic motivation of users, all over the generation, provisioning and control stack by acquiring immediate feedback on society state.

Moreover, a large-scale distributed management system is required that can process huge amounts of event data and operate in real time. It should be able to manage the interface with infrastructures such as service market platforms that support the cooperation of various players. It, thus, helps to automatically balance highly fluctuating supply and demand, in a reliable and cost-effective manner. Relying on crowd sourcing [31] in a ubiquitous society, we can obtain needed services by soliciting contributions from the society rather than from traditional suppliers.

5.2.1 CoE Architecture

CoE is inspired by the idea of federating ubiquitous P2P network platform and the classic distributed data centers to form a multi-layer interactive architecture. CoE fulfils hierarchical control system goals in the integrated system of XaaS that supports both computing and energy service provisioning.

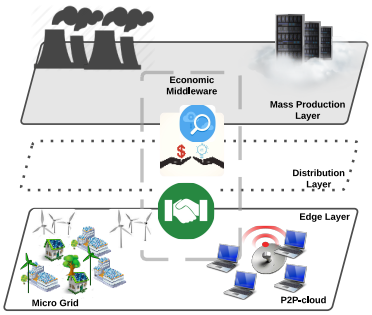


FIGURE 5.1: CoE Architecture

CoE offers establishing Virtual Power Plant (VPP) and Virtual Private Cloud (VPC) for each vicinity through the local broker. VPP leverages existing grid networks to tailor electricity supply and demand services for a customer. VPP maximizes value for both the end user and the distribution utility using a set of software-based dynamic systems to deliver value in real time, and can react quickly to changing customer load conditions.

All the same, VPC is a cost-effective solution to expand the presence into the public cloud instead of expanding private infrastructure. With its pool of highly available compute, storage, and networking resources, VPC fits well in scenarios involving variable or bursting workloads, test and development, and next generation mobile applications.

In CoE, there is a pool of providers, i.e. energy and computing service providers, including prosumers in the edge layer, and mass producers in the higher layer. CoE layered architecture assures quality of service via improving resource availability in the edge-layer by the support from the mass production layer. A layered architecture of CoE is illustrated in Figure 5.1. Horizontal layers represent a hierarchical division of the service providers. Prosumers, i.e. consumers and retail service producers, at the bottom layer constitute the edge layer locally under the concept of vicinity, as illustrated in Figure 5.2. Classic cloud service providers and mass energy providers are categorized as the mass service providers in the highest level. The lower layers promote energy efficiency in resources usage and

the employment of greener sources of energy. Meanwhile, the higher levels can ensure resource availability and cope with power variations in the edge-layer power output.

In the CoE architecture, hierarchical brokers are responsible for managing the market in different layers. These brokers are cross layer agents that are in charge of hosting auctions and providing feedback to the layers below and above, in the economic middleware, as demonstrated in Figure 5.2.

In this architecture, there is a bidirectional information flow. While wholesale brokers are statically placed, local controller/broker agents can be dynamically placed in any prosumer location providing that the prosumer can obtain the computing and energy requirements for the broker. In broker placement, the priority is with the source which has excess energy generated. Dynamic local controller placement contributes to the energy efficient data processing and movement, which is the key for a sustainable system.

5.2.2 Agent Based CoE Service Composition

The CoE platform, as illustrated in Figure 5.1, can be modeled with the concept of multi-agents, since the autonomous resource agents are distributed all through the edge layer as well as the hierarchy of the system. Agents are autonomous computing and/or electricity prosumers (producer/consumer) systems that are capable of making decisions independently and interacting with the other agents through cooperation by working together and drawing on each other's knowledge and capabilities. They can achieve the state in which their actions fit in well with the others via coordination, or negotiate to reach agreements on some matters [120].

The multi-agent system is the most suitable platform to model the distributed collaborative system requirements based on its properties and functionality, allowing it to implement intelligence in the smart grid control due to its social ability, flexibility, self-healing features and economic agent support [132]. Moreover,

agent-oriented computing provides a natural paradigm for automating the interactions among complex interconnected systems. Therefore, we frame the economic middleware conforming agent-oriented architecture, with the following description.

5.2.2.1 Environment

In CoE, we have nested environments through the hierarchy of the architecture, which amount to a set of producers and consumers, and brokers. Looking closer, prosumers make a rich, heterogeneous environment which is controlled by coordinators, in order to drive the prosumers behavior and represent the interest of a group of prosumers on the market.

5.2.2.2 Agents

Agents include prosumers, brokers in different levels, mass producers of electricity and cloud services (in the mass production layer). **Prosumer** agents produce services in the retail level and are the end users of the services, at the same time. Each prosumer is equipped with a cloud and electricity controller, to regulate and control its demand and supply. **Broker** agents in different layers can decide what strategies to employ both on the market and prosumers. Each **local broker** is authorized to run its own market regulation mechanism to supply the demands locally, as long as it does not violate the wholesale market's framework. This elevates the decentralization, better scalability and speed of adjustment to varying local conditions, while bounding global imbalances. Utility and cloud service providers can trade the mass provider services on their behalf via the **mass production broker**.

5.2.2.3 Market Rules

Since energy and computer systems provide two different services, to integrate these two systems, in our market model, we need a metric that can measure the

contribution of each service in an understandable scale for the other. Moreover, a universal metric facilitates the collaboration of the two systems. Virtual money seems to be an appropriate metric for this end.

1. *Local Currency:* Defining local currency in the micro-grid-community level can incentivize the users to collaborate in the system by sharing the resources, i.e. energy and computing by earning credits. The idea behind defining a local currency is to drive and improve the coordination of users within a vicinity and promote the vicinity among the others by elevating the value of their local currency against them. Moreover, this mechanism helps in load balancing by changing the value of local currency, by allowing arbitration.

2. *Redeeming the value for idle resources:* When local supply exceeds the local demand, the local broker can assign bitcoin [93] generation tasks to the prosumeres offering resources, in exchange of certain amount of local currency based credit in their account. Therefore, the *available resources are not effectively lost and can be re-acquired later from mass producers*, if supply is scarce in the vicinity. This is specially useful when the *energy powering the idle resources is green energy that is being under-utilized*. Thus, we can in a novel way, *effectively attempt at preserving resources and energy as effective reserves for later demand*.

3. *Resource provisioning from outside the vicinity:* Local brokers, to provide resources from outside the vicinity, can only rely on some outside currency, i.e. the bitcoin generated in the vicinity when there are excess resources of electricity and computing in the vicinity (as an ideal universal replacement to any legal tender or precious metal). Afterwards, to deliver the service to the end user, local broker charges the users based on the local currency value equivalent to the amount of bitcoin and the associated conversion taxes.

4. *Pricing mechanism:* In order to encourage the agents to provide resources locally, an adaptive tax rate is defined. Tax is applied to the services provided

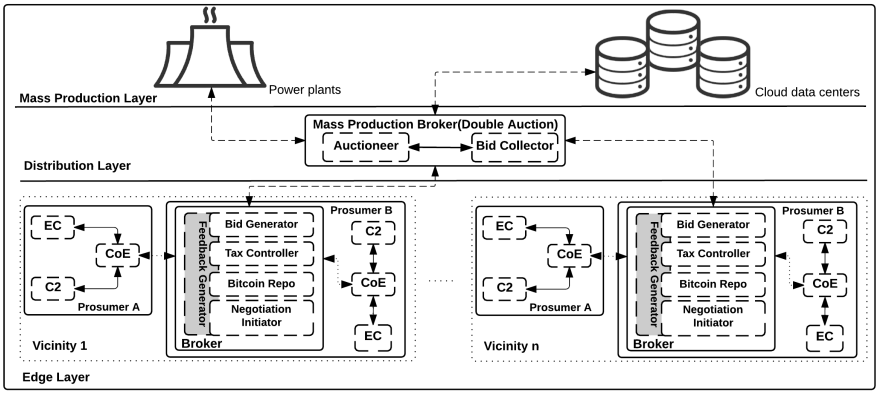


FIGURE 5.2: ARTA Middleware Architecture

from outside the vicinity. Therefore, users are incentivized to acquire their resources from inside the vicinity. However, to preserve the service quality, in case of resource scarcity or unreliability due to system dynamics and uncertainties, the tax rate is decreased.

5.3 Economic Middleware

An Economic Middleware acts as an interface to facilitate smart electricity and computing service trading across the CoE hierarchy and horizontally in the edge layer. Here, we introduce ARTA (Agent-oriented Resource Trading Architecture) middleware, which follows the agent-oriented architecture.

5.3.1 Middleware Components

ARTA middleware, as shown in Figure 5.2, includes the following components:

- **Energy Controller(EC):** This module exists in each prosumer side, and is able to predict and measure the energy consumption of each appliance at home.

- **Computing Controller(C2):** In each prosumer of the edge layer, C2 plays the same role of EC for the computing services.

- **Local broker:** It is responsible for defining the tax rate based on the difference of the demand and local supply it receives. If the demand and supply do not match and the vicinity encounters resource scarcity, the broker decreases tax rate, through **tax controller**, to make the external resources more affordable for the end users. Moreover, the broker should submit the bids to the higher level broker, to obtain the resources for excess demand of the vicinity. This task is performend in the **bid generator** module. A **bitcoin repository** component is responsible to keep the bitcoin balance of the vicinity which is necessary for trading with mass production broker, in the outside world. Bitcoin [93] is an online payment system, in which trade parties can transact directly without the interference of any intermediary, through bitcoin. Prosumers submit all the demands and the offered resources to the local broker. It is the responsibility of the broker to find the matching supply and demands; thus, initiate the negotiation between them, by proposing the list of tentative matching prosumer agents. Besides, local broker provides feedback to the different modules inside it, via collecting data from each module and generating feedback by means of **feedback generator**. Some feedback about the system overview is also sent to the prosumers in the vicinity to cover the partial view that each prosumer sketches from the system.

- **Mass production broker:** This agent is in charge of collecting bids and setting up auctions among different service providers of mass production layer for the demands submitted by the local brokers. It is composed of two major components of **bid collector** and **auctioneer**.

5.3.2 Middleware Specifications

ARTA should cope with *highly dynamic* and *perishable* resources. Computing and energy resources available in each moment are not storeable; hence, resource

capacities not utilized now are worthless in the next moment. We cannot store the computing capacity when it is idle to exert it later when needed. Besides, the overhead of keeping the idle resources standby or switching the idle capacity to sleep mode in terms of energy and latency, makes the model to design some mechanisms to address this issue. ARTA tackles this problem by aligning the demand to the available resources. Also it makes redeeming the idle resources possible by generating bitcoin in idle resources. This can be enacted by a bitcoin mining pool coordinated by ARTA.

Moreover, in the edge layer of the system, resources are volatile due to the system uncertainties. Uncertainty is rooted in the prosumers' natural behavior, as long as they can choose to contribute to the system or leave it. Moreover, the unpredictability in the renewable energy production aggravates the problem. Therefore, the economic middleware should be robust enough to cope with the dynamic resource availability. In ARTA negotiating agents get feedback on system dynamics and consider it in their decisions, as explained in Section 5.4.

5.3.3 ARTA's Requirements

An economic middleware enabling the resource trading in the CoE system should address the following CoE system requirements. It should follow load balancing and reliability goals while keeping system's resiliency and sustainability in perspective. In this section, we elaborate on these requirements.

- **Reliability:** Further to specific resource management in a highly dynamic system, ARTA should be reliable by covering the fault tolerance, error resiliency and provisioning a desired level of trust in the resource trading. To reinforce the fault tolerance of the distributed system, we store the data in distributed data storage accessible to all the prosumer agents in the vicinity if they have access to the token.

- **Load balancing:** By using market mechanisms, we can regulate supply and demand. ARTA follows this goal by decreasing the gap between demand and supply patterns, through exerting vicinity sizing and adopting the negotiation and auction strategies proposed in Economics, as discussed in Section 5.4. Dynamically changing the vicinity size empowers aligning the demand and supply by regulating the resource availability. Moreover, ARTA applies dynamic exchange tax rate according to the resource availability.

5.4 Economic Model

ARTA offers a bi-level resource provisioning strategy which includes two protocols, each following a different economic model, yet, they can co-operate with each other to improve the ARTA's performance and scalable implementation.

A negotiation based protocol (*horizontal protocol*) is provided to exchange the resources in the edge layer, i.e. locally in each vicinity. The prosumer agents negotiate directly to supplier and provide the resources locally. This negotiation is initiated by the local broker of the vicinity and goes on through the direct communication of the negotiation agents.

Moreover, to acquire the resources externally from the mass production layer via the distribution layer broker, a double auction mechanism is exerted to devise a *vertical protocol*. In the rest of this section, we portray the protocols and mechanisms designed to form the economic model.

5.4.1 Horizontal Negotiation Protocol

This protocol is run in the edge layer, to facilitate the Prosumer-to-Prosumer (P2P) negotiation. Negotiation between each pair of prosumers is performed by making proposals in iterative rounds until either an agreement is reached or at least one of the negotiating agents misses the deadline. The service deadline is defined according to the service flexibility/availability.

Note that this negotiation can be many-to-many due to the multilateral nature of the CoE system. Videlicet, each prosumer agent can negotiate deals with multiple prosumers simultaneously.

Negotiation Policy: Each prosumer considers the following specifications and quantifies them according to its desires, conforming the strategies formulated here.

- **Initial Price** The initial price identifies the most reasonably desirable price that an agent is willing to sell or buy the services. Each agent defines its initial price P_i^0, independently, and according to the system feedback.
- **Reserved Price** Indicates the max/min price that an agent inclines to exchange the money/resources, P_i^r.
- **Service Deadline** By defining a time dependent bargaining strategy, an agent considers the deadline in the negotiation. d_i stands for the service/resource deadline. The closer the service is to the deadline, the faster the negotiation price converges to the reserved price.
- **System Dynamics** λ coefficient is defined to pace the negotiation according to the service specifications as well as system dynamics. In order to consider the system dynamics, we define a feedback presenter $\mathfrak{s}(\mathfrak{t})$ in the range of [0-1] which is sent regularly to the prosumers from the broker. The lower the value of $\mathfrak{s}(\mathfrak{t})$, the more dynamic the system. $\mathfrak{s}(\mathfrak{t})$ is computed in the feedback generator following (5.1) and sent to the prosumers in each round exploiting a gossip protocol [72].

  The probability of gossip exchange is defined according to the entropy of the information to transmit, $P_{forward}(t) = 1-|s(t)-s(t-1)|$. The bigger the gap between $\mathfrak{s}(\mathfrak{t})$ and $\mathfrak{s}(\mathfrak{t}-\mathfrak{1})$, the more effort is required to update the feedback throughout the vicinity. Hence, the update is triggered in all nodes, only if the system state is significantly changed compared to the previous round. This way, we can control the system overhead. Loosely paraphrasing, if there is little change in the $\mathfrak{s}(\mathfrak{t})$ value, relying on the previous value, $\mathfrak{s}(\mathfrak{t}-\mathfrak{1})$ does not make remarkable changes in the agents decision. Therefore, they

can remain using the previous feedback, and save message exchanges in the network. By default, if a prosumer does not receive feedback in round t, it replaces $s(t)$ by $s(t-1)$.

$$\mathfrak{s}(\mathfrak{t}) = \frac{1}{2}(\frac{n(t)}{\bar{n}} + \frac{max(0, n(t) - n(t-1))}{n(t)}) \tag{5.1}$$

In (5.1), $n(t)$ represents the number of the prosumers connected to the vicinity at time t. $\bar{n}$ denotes the average number of prosumers over time, so far. Thus, $s(t)$ is calculated based on the normalized ratio of the number of the prosumers in the vicinity to the average number of prosumers over time, and the difference of the currently available prosumers and the prosumers in the previous round, if the number is decreased.

- **Market Perception** β_i is computed by each agent i to evaluate its price proposals compared to the offers it receives from the provider agents, it is negotiating with. Market perception is calculated using (5.2). In this formula, the average ratio of immediate changes in the providers proposals to the average proposal change over time is considered to estimate the market perception of each consumer agent i, which is in negotiation with $|providers|$ provider concurrently. Note that each consumer agent has only partial view of the system and calculates the market perception accordingly.

$$\beta_i(t) = \frac{1}{|providers|} \sum_{j \in \{1,2,\ldots,|provider|\}} \frac{P_j(t-2) - P_j(t-1)}{\frac{P_j^0 - P_j(t-1)}{t}} \tag{5.2}$$

Negotiation Strategy: During the negotiation, each consumer proposes a price which varies in each round. At first, it starts by P^0 and adopts the value according to the service deadline and system dynamics, as depicted in (5.3).

$$P_i(t) = P_i^0 + (\frac{t}{d_i})^{\lambda_i(t)}(P_i^r - P_i^0) \tag{5.3}$$

In (5.3), t denotes the time elapsed from the beginning of the negotiation, and d_i is the service deadline. Term $\frac{t}{d_i}$ represents the time left for providing the

$$\lambda_i(t) = \begin{cases} \lambda_i(t-1) + \beta_i(t-1) + \mathfrak{s}(t) & \lambda_i(t-1) > 1 \\ \max(\lambda_i^{min}, \lambda_i(t-1) + (\beta_i(t-1) + \mathfrak{s}(t))(\lambda_i(t-1) - \lambda_i^{min})) & 0 < \lambda_i(t-1) \leq 1 \end{cases} \tag{5.4}$$

resource; thus, the closer the deadline, the higher the increase in the proposed price. Further to the service deadline, system dynamics and market perception affect the proposed price. Therefore, λ coefficient should be adapted dynamically during the negotiation, as shown in (5.4).

λ_i^{min} is applied to avoid unnecessarily incurring in utility losses, due to too rapid conceding. The rational behind this strategy to choose the $\lambda_i(t)$ coefficient is that a consumer can tune its proposals according to the market perception. Looking closer, loosing the market position according to the perception leads to faster λ adjustment, while being more stable in the market (increasing $\beta_i(t)$) results in slower changes in $\lambda_i(t)$. All the same, the more stable the system is, the higher the probability of finding a resource provider for negotiation; therefore, the slower changes in $\lambda_i(t)$ value. In contrast, in less stable system situation, the negotiation should be finished before the resources disappear from the system; hence, the convergence to the P_i^r happens more quickly.

Producer Side Negotiation Strategy: Producer follows the same strategy as the consumer in the system, but it decreases the proposed price instead of increasing it, i.e. $P_j^0 \geq P_j(t) \geq P_j^r$. Thus, the pricing follows the formula below.

$$P_j(t) = P_j^0 - (\frac{t}{d_j})^{\lambda_j(t)}(P_j^0 - P_j^r) \tag{5.5}$$

Each prosumer can decide about its own prices in each round of auction according to the exponential increase/decrease policy indicated as λ, which is in range of zero to one. The closer the deadline to provide the service (d), the higher/lower the requester's/supplier's proposed price will be. P^0 represents the initial price, and P^r/P^0 is the maximum and minimum price to offer, respectively.

5.4.2 Vertical Auction Protocol

In the distribution layer broker, as visualized in Figure 5.2, a double auction module exists. A double auction mechanism facilitates the resource exchange between mass production layer and the consumers at the edge of the system, pursuing the traditional utility providers' model.

Biding: Every edge-layer and mass production layer provider offers a bid to the distribution layer broker. The bid from each agent A_i is in the form of a tuple b_i with four entries, resource type θ_i, resource amount q_i, the price per unit resource p_i and a flag $\mathfrak{f}_i$ that indicates if the agent provides the resource or requests it. Note that each agent can only submit one bid of the same resource at each round $\forall_{i,j}\theta_i = \theta_j \Rightarrow b_i = b_j$; otherwise, all the bids for the same resource type from that agent are deleted before running the auction step. Moreover, to help preventing malicious bids, p_i should be greater than a pre-defined threshold, otherwise the bid is rejected before running the auction.

Auction: The auction follows the sealed price mechanism; therefore, each participant is only aware of its own proposal. After collecting all bids, the distribution layer broker sorts all the bids according to the proposed prices ascending for the providers and descendingly for the requesters. Then a double auction runs in the auctioneer module and the results are announced to the participants. All the unsuccessful bids should be revised and resubmitted in the next round. In the next section, we assess the feasibility of the proposed CoE system.

5.5 Evaluation

We study the benefits of rolling out the CoE and elaborate the feasibility of the proposed architecture by answering several questions across this section. We follow the same experiment setup as the previous chapters.

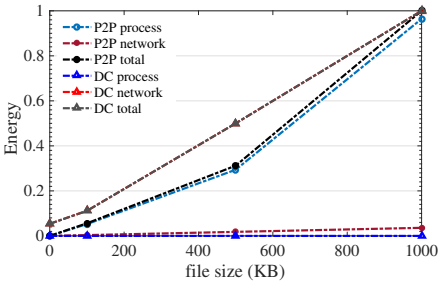


FIGURE 5.3: Energy consumption for sending different data sizes

5.5.1 Leveraging P2P-cloud for Smart Grid Data Processing

Here, we show how leveraging the P2P-assisted cloud as the communication and computing platform for data aggregation can improve the energy efficiency and service price as a consequence. In this section, we elaborate on the energy consumption of data aggregation, for different data sizes on data center and P2P-cloud, which are natural candidate computing platforms for the smart grid.

To estimate the energy consumption per read and write process of different file sizes, we exploit the disk latency data from [85] on Clommunity [3] testbed. For estimating the energy we rely on the formula introduced in Chapter 3.

The energy depleted for storing and transmitting the data sizes over P2P-cloud and data centre is depicted in Fig.5.3. The energy consumption of P2P-cloud network is remarkably less than that of the data center communication, since cloud relies on the Internet to interact with the end users and the Internet infrastructure is utilized with high power devices. The storing energy, however, is marginally lower for the data centers. Here, we study just the case that P2P-cloud relies on the local wireless antennas for communication. Provisioning the services of P2P-cloud through non-local providers draws almost the same energy as the data centre, since it relies on the Internet infrastructure to communicate with non-local nodes.

[3] `http://wiki.clommunity-project.eu/`

TABLE 5.2: Energy consumption in bi-level aggregation

	Communication	**Process**	**Overall**
Data center	99.99	10^{-6}	100
Bi-level	0.99	10^{-9}	1

Therefore, mechanisms prioritizing the local providers are more energy and cost efficient.

However, in some cases, P2P-cloud cannot obtain the computing or storage resource required for a large amount of data. In such cases, we propose bi-level aggregation in which sensor data conveys to the community storage. Afterwards, the stored data are aggregated and sent to the cloud for further processing. Exploiting this method, we can save a conspicuous amount of energy.

Table 5.2 illustrates this difference in comparison with directly sending the raw sensor data to the data center. As we see in this table, the energy needed to transmit data directly to the data center is 100 times more than the energy required for bi-level data aggregation.

5.5.2 Is bi-level architecture able to incentivize the collaboration?

Defining cost as the main incentive, CoE can improve the collaboration among the prosumers, through the credit earning mechanism. Figure 5.4 illustrates that more resources are provided within the vicinity in CoE compared to the random resource allocation mechanism.

Here, we only consider flexible service provisioning in the edge to assure the quality of service due to the uncertainty of renewable retail generators. Both electricity and computing services can be classified as rigid and flexible. While rigid service needs real time resource provisioning, flexible services can be scheduled for a later time, and is more flexible.

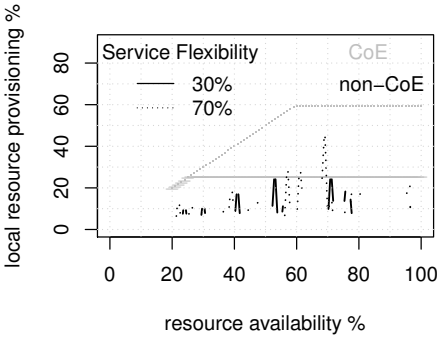


FIGURE 5.4: Collaboration

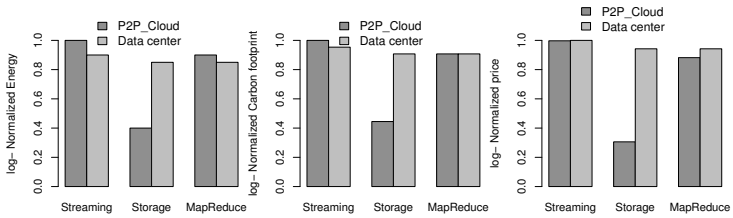
As illustrated in Figure 5.4, local resource provisioning depends on service flexibility and resource availability in the vicinity. Here we studied two service models, with 30% and 70% flexibility. The results show that the more resources available in the vicinity, the higher collaboration of prosumers occur in CoE compare to the random collaboration case. The collaboration in non-CoE case, however, is weekly correlated to the resource availability in the vicinity. Note that in CoE we do not consider the possibility of the inter-vicinity collaboration, since there is a significant transmission loss and quality of service degradation in this case.

Implication 1: *Increasing the resource availability at the edge layer of the CoE should be considered as a priority to attain the smart grid objectives.*

5.5.3 How much energy can be saved in CoE?

Figure 5.5.a depicts how much energy can be saved by smart service provisioning in CoE[4]. We see that some cloud services such as storage as a service in the P2P-cloud, i.e. edge layer, is more energy efficient compare to the data center case, while two other services are better to obtain via data centers in the higher

[4]Experiment setup is the same as what we described in the previous chapter.



a) Energy consumption b) Carbon emission c) Price

FIGURE 5.5: Energy, Carbon and Price of different services

layer. Therefore, combination of edge devices and data centers result in a more energy efficient services providing that resources are allocated in an energy efficient manner. For this end, a framework is required to characterize the energy efficacy of each individual service in both platforms. A decision support system can help afterwards according to the analysis results.

Increasingly, in Figure 5.5.b, the carbon emission of different services are compared. We assume that the prosumers are equipped with the solar roof tops, which emit 41 g/kWh and data centers equipped with 50% of renewable solar energy produced by solar PV at utility level and generate 48 g/kWh of CO2 in average and 50% of brown energy inducing 802 g/kWh of carbon footprint in an average case, according to [54]. This figure reveals the fact that, carbon emission as an incentive, besides energy consumption may turn the table in more cases in favor of P2P-cloud, due to the lower emission rate of prosumer level renewable energy generators.

Implication 2: *carbon emission rate is a better metric than energy consumption to quantify the efficacy of the system in fulfilling smart grid objectives.*

5.5.4 How much cost will be saved?

In the state-of-the-art mechanisms, computer services are priced regardless of the energy consumption cost. However, energy aware service provisioning can save remarkably in the provider costs, since energy is a major part of dynamic price in

the cloud service provisioning. Exerting CoE, we have a better chance of directing services to the appropriate layer of provisioning, and saving energy cost as a consequence.

Besides, CoE provides an opportunity to share the infrastructure and data required in smart grid and cloud instead of duplicating the resources. Namely, in case of carbon based charging, finding a cheap energy source will be significantly important. In such a case, being aware of renewable energy sources can help saving in dynamic cost. CoE as an integrated architecture will obtain the smart grid data to the brokers across the hierarchy, instead of duplicating this data in two separate systems of cloud and smart grid.

Figure 5.5.c illustrates the cost of energy in a carbon based energy pricing, which assigns the same price to all the energy sources and applies carbon taxes according to the carbon-footprint portion attributed to the electricity source. As shown in this figure, in all cases, P2P-cloud service provisioning leads to cost saving. Nevertheless, we should bare in mind that there is limited resource availability for local resource provisioning and the quality of service may not be obtained in local service providing.

Implication 3: *carbon footprint is cleaner metric for energy pricing; however, it is not straight forward to move from energy based pricing to carbon based pricing since the latter one fails to handle peak demand management as well as energy based pricing. Therefore, a combined energy pricing method is more desirable.*

5.5.5 Is implementation complexity warranted?

CoE reveals that integration facilitates a diverse range of service exchange. However, integration may incur more complexity to the economic layer in the system due to the different nature of each system such as uncertainty level, storablity, flow management complexity, etc. This added complexity should be warranted with the advantages of integration, e.g. more effective marketplace. To attain

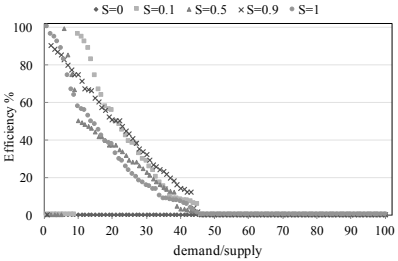


FIGURE 5.6: Negotiation Protocol Efficiency

CoE goals, we need a robust economic model which can manage the demand and supply in a multi-variable marketplace.

Nonetheless, if we aim at greening the ICT while exerting ICT for green, CoE can be a good candidate to reduce carbon emission, save energy and cost as a consequence of smart service provisioning.

5.5.6 Economic Model Efficiency

Efficiency of the economic model is defined as the amount of the demand served inside the vicintiy. As depicted in Figure 5.6, with the demand to supply ratio in the range of 10%-45% ARTA performs more efficiently due to the enough resource availability in the system. The success rate of the negotiations in this range is over 90%. However, increasing the demand to supply ratio leads to resource scarcity and therefore, the model cannot work efficiently in those cases. In demand to supply ratio of below 10%, efficiency is very low, because of underutilized resources. Moreover, as shown in Figure 5.6, the more stable the system, the higher the gained efficiency.

5.5.7 Negotiation Protocol Overhead

As stated, to tackle system dynamics in each round of the negotiation, the system stability parameter is updated proactively. However, to prevent the redundancy

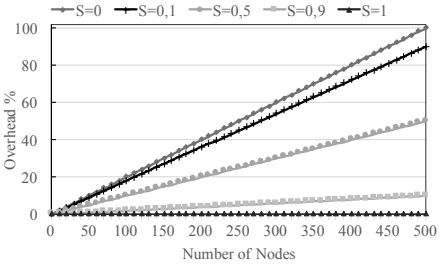


FIGURE 5.7: Negotiation Protocol Scalability

which leads to excess overhead in the system, we applied a gossip control mechanism that exchanges the messages according to the entropy of the information. Therefore, the more stable the network, the fewer messages the system needs to exchange. Figure 5.7 confirms this statement by producing zero overhead when the system is perfectly stable, i.e. $\mathfrak{s}(\mathfrak{t}) = 1$. This protocol runs the proactive updating with reactive overhead. Therefore, even in the worst case ($\mathfrak{s}(\mathfrak{t}) = 0$), the overhead grows linearly by increasing the vicinity dimensions and remains under 20%, in the vicinity of smaller than 100 nodes.

5.5.8 Middleware Scalability

As illustrated in Figure 5.7, in case that the system does not experience high resource dynamicity, i.e. $S \geq 0.5$, the overhead of middleware maintenance increases linearly by increasing the vicinity size, but it is still below 50%. A very stable system confronts overhead of less than 10% in a very large vicinity. However, we notice that even in the case of a highly dynamic system, a vicinity of 100 nodes can survive by producing the overhead of less than 20%.

Implication: Shrinking the vicinity size in highly dynamic situations improves the system sustainability.

5.6 Summary

In this chapter we introduced Cloud of Energy (CoE). CoE envisages the service provisioning framework of the future that provides everything as a service via an integrated cloud and smart electricity grid platform in horizontal and vertical dimensions. CoE facilitates the resource management in each of smart grid and cloud through their hierarchy. It also expedites the horizontal integration of different services via their shared economic incentives.

The economic layer acts as a middleware to translate a service in every concept, e.g. energy and computing, to the common incentive scale of money. Integration elevates the collaboration of diverse range of providers and consumers, requesting for different services. Moreover, an integrated system is more efficient and greener, since it avoids unnecessary redundancy in the common sub-systems, such as shared data, computing and communication infrastructure, etc. Also the integration leads to greener system since it provides increased energy awareness.

Part IV

Closure

—"*Success is not final; failure is not fatal.*
It is the courage to continue that counts."

-Winston Churchill

6

Conclusion

This chapter closes the dissertation by reviewing the whole work presented so far. Moreover, it introduces some aspects of on-going and future work.

6.1 Put It All Together

We recall the goals and challenges that motivated this work by addressing two major questions enumerated as follow.

First we need to find out if it is energy efficient to move toward P2P-clouds. Addressing this question requires a framework to compare energy consumption for each service. Nonetheless, this analysis framework may be trapped in decisions that incur in significant performance degradation. Therefore, a performance aware

energy analytic metric is needed to tackle this issue. To address this, we introduced the energy effectiveness metric [113], in Chapter 3, conceptualized to the energy and performance requirements of each layer across the service provisioning stack, i.e. application, VM/OS, and hardware. Energy effectiveness can assess how successful the ecosystem is from a particular perspective based on different granularity of information.

Our studies in [113, 117, 118] reveal that there is no straight forward answer to this question, since the answer not only depends on the service specifications, but also partially depends on the hardware setting, where the service is running on. Therefore, to answer this question, we need a framework to analyze energy consumption of a service across different platforms, as sketched in Chapter 4.

Moreover, we introduced the idea of Cloud of Energy in Chapter 5, to move toward a more sustainable ecosystem. Energy and Information and Communication Technology (ICT), as two driving forces of the contemporary life, are reshaping themselves based on ubiquitous society architecture to improve their service quality. Within the reforming process, integration of two systems can contribute to a greener ubiquitous society by equipping them with the concept of energy conservativeness, and leveraging renewable energy sources.

In this work, with the idea of Cloud of Energy (CoE) [115] outlined, we foster the adoption of green energy and green cloud by integrating these two systems. CoE introduces an integrated framework of everything as a service to facilitate the service exchange, not only across the computing and electricity grid hierarchy, but also among them via an economic middleware [116].

6.2 Concluding Remarks

Energy awareness in literature has been considered to contribute to two different goals of energy cost saving and energy sustainability. In this dissertation, we addressed energy sustainability by following two major objectives: i) energy

saving and ii) fostering the adoption of green energy. We believe that these two objectives can redefine the energy cost saving concept, whereas they cover energy conservativeness and using renewable energy sources as the bases of energy cost saving.

Nonetheless, energy accounting policies could also be updated to incentivize these objectives through charging lesser for renewable energy sources compared to classic gray/brown energies. A solution could be applying carbon footprint taxes.

Another implication of this dissertation is the impact of the resource availability on the overall energy consumption in the ecosystem. Albeit, the classic vantage point encourages equipping the systems with more resources to increase the service quality, our study reveals that elevating service quality through enhancing resource availability is not a panacea solution. Our findings indicate that increasing hardware resources leads to more static power dissipation and may result in added complexity in system administration.

Namely, in wireless infrastructure, more resources provoke the interference effect and power dissipation to cope with the interference. Therefore, in forming the P2P-assisted cloud architecture, a dynamic vicinity sizing mechanism, considering the above issue, is vital to move toward energy efficiency goals.

6.3 Future Work

The work presented in this dissertation motivates the following future research directions:

A line of work can be proposing a distributed decision making system to decentralize the scheduling and resource allocation policy introduced in the energy analysis framework, integrated with OpenStack. Moreover, this resource allocation policy can contribute to a more accurate energy-aware user accounting in virtualized and multi-tenant ecosystems. Formalizing the VM energy consumption, integrated

with more accurate hardware power models, facilitates fair energy-aware VM pricing.

Moreover, as mentioned, embedding cache in a P2P-cloud architecture helps in improving the energy efficiency of such an ecosystem. We can further increase the energy efficiency of the system by applying energy efficient caching as well. Energy efficiency of caching can be reinforced via considering the popularity of content and hit ratio in caching algorithms.

Considering CoE architecture, in the edge layer of the system, vicinity size plays a significant role in promoting the energy efficiency within the vicinity. As stated, increasing the resource availability leads to more static power dissipation. Thus, specially in case of under-utilized resources, where excess resources induce energy non-efficiency in the system, resizing the vicinity dynamically can contribute to increasing the overall energy efficiency. Dynamic vicinity sizing should contemplate the demand and uncertainties in the system to obtain the resources efficiency. Conjointly with the techniques offered to align the demand with the supply, vicinity sizing can decrease the demand and supply gap by regulating the supply according to the demand.

As future work in line with our contribution in introducing the economic middleware in CoE, we can envisage covering system security and define more clear fault tolerance mechanism.

References

[1] Green space, http://www.greenpeace.org/international/global/international/planet-2/report/2010/3/make-it-green-cloud-computing.pdf.

[2] Peer energy cloud, http://www.peerenergycloud.de/.

[3] Revolutonizing network design flattening the data center network with the qfabric architecture, http://www.itbiz.com.ua/media/docs/juniper/qfx/the%20q%fabric%20architecture.pdf.

[4] Abts, D., Marty, M. R., Wells, P. M., Klausler, P., and Liu, H. (2010). Energy proportional datacenter networks. In *ACM SIGARCH Computer Architecture News*, volume 38, pages 338–347. ACM.

[5] Al-Fares, M., Loukissas, A., and Vahdat, A. (2008). A scalable, commodity data center network architecture. In *ACM SIGCOMM Computer Communication Review*, volume 38, pages 63–74. ACM.

[6] Alicherry, M. and Lakshman, T. (2012). Network aware resource allocation in distributed clouds. In *INFOCOM, 2012 Proceedings IEEE*, pages 963–971. IEEE.

[7] Amsel, N., Ibrahim, Z., Malik, A., and Tomlinson, B. (2011). Toward sustainable software engineering (nier track). In *Proceedings of the 33rd International Conference on Software Engineering*, pages 976–979. ACM.

[8] Ardito, L., Procaccianti, G., Torchiano, M., and Migliore, G. (2013). Profiling power consumption on mobile devices. *IARIA*.

[9] Baliga, J., Ayre, R. W., Hinton, K., and Tucker, R. (2011). Green cloud computing: Balancing energy in processing, storage, and transport. *Proceedings of the IEEE*, 99(1):149–167.

[10] Baliga, J., Hinton, K., and Tucker, R. S. (2007). Energy consumption of the internet. In *Optical Internet, 2007 and the 2007 32nd Australian Conference on Optical Fibre Technology. COIN-ACOFT 2007. Joint International Conference on*, pages 1–3. IEEE.

[11] Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., and Warfield, A. (2003). Xen and the art of virtualization. *ACM SIGOPS Operating Systems Review*, 37(5):164–177.

[12] Barroso, L. A. and Hölzle, U. (2007a). The case for energy-proportional computing. *Computer*, (12):33–37.

[13] Barroso, L. A. and Hölzle, U. (2007b). The case for energy-proportional computing. *IEEE computer*, 40(12):33–37.

[14] Belady, C., Rawson, A., Pfleuger, J., and Cader, T. (2008). Green grid data center power efficiency metrics: Pue and dcie. Technical report, Technical report, Green Grid.

[15] Bellosa, F. (2000). The benefits of event-driven energy accounting in power-sensitive systems. In *Proceedings of the 9th workshop on ACM SIGOPS European workshop: beyond the PC: new challenges for the operating system*, pages 37–42. ACM.

[16] Beloglazov, A. and Buyya, R. (2010). Energy efficient resource management in virtualized cloud data centers. In *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, CCGRID '10, pages 826–831.

[17] Beloglazov, A. and Buyya, R. (2012). Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. *Concurrency and Computation: Practice and Experience*, 24(13):1397–1420.

[18] Beloglazov, A., Buyya, R., Lee, Y. C., Zomaya, A., et al. (2011). A taxonomy and survey of energy-efficient data centers and cloud computing systems. *Advances in Computers*, 82(2):47–111.

[19] Bender, M. A., Brodal, G. S., Fagerberg, R., Jacob, R., and Vicari, E. (2010). Optimal sparse matrix dense vector multiplication in the i/o-model. *Theory of Computing Systems*, 47(4):934–962.

[20] Berral, J. L., Goiri, Í., Nou, R., Julià, F., Guitart, J., Gavaldà, R., and Torres, J. (2010). Towards energy-aware scheduling in data centers using machine learning. In *Proceedings of the 1st International Conference on energy-Efficient Computing and Networking*, pages 215–224. ACM.

[21] Bertran, R., Becerra, Y., Carrera, D., Beltran, V., Gonzàlez, M., Martorell, X., Navarro, N., Torres, J., and Ayguadé, E. (2012a). Energy accounting for shared virtualized environments under dvfs using pmc-based power models. *Future Generation Computer Systems*, 28(2):457–468.

[22] Bertran, R., Buyuktosunoglu, A., Gupta, M. S., Gonzàlez, M., and Bose, P. (2012b). Systematic energy characterization of cmp/smt processor systems via automated micro-benchmarks. In *Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 199–211. IEEE Computer Society.

[23] Bertran, R., Gonzàlez, M., Martorell, X., Navarro, N., and Ayguadé, E. (2013). Counter-based power modeling methods: Top-down vs. bottom-up. *The Computer Journal*, 56(2):198–213.

[24] Bila, N., de Lara, E., Joshi, K., Lagar-Cavilla, H. A., Hiltunen, M., and Satyanarayanan, M. (2012). Jettison: efficient idle desktop consolidation with

partial vm migration. In *Proceedings of the 7th ACM european conference on Computer Systems*, pages 211–224. ACM.

[25] Bilal, K., Malik, S. U. R., Khalid, O., Hameed, A., Alvarez, E., Wijaysekara, V., Irfan, R., Shrestha, S., Dwivedy, D., Ali, M., et al. (2013). A taxonomy and survey on green data center networks. *Future Generation Computer Systems*.

[26] Bircher, W. L. and John, L. K. (2007). Complete system power estimation: A trickle-down approach based on performance events. In *Performance Analysis of Systems & Software, 2007. ISPASS 2007. IEEE International Symposium on*, pages 158–168. IEEE.

[27] Birman, K. P., Ganesh, L., and van Renesse, R. (2010). Running smart grid control software on cloud computing architectures. *Workshop on Computational Needs for the Next Generation Electric Grid*.

[28] Bohra, A. E. and Chaudhary, V. (2010). Vmeter: Power modelling for virtualized clouds. In *Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on*, pages 1–8. Ieee.

[29] Bonomi, F., Milito, R., Zhu, J., and Addepalli, S. (2012). Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16. ACM.

[30] Borthakur, D. (2007). The hadoop distributed file system: Architecture and design. *Hadoop Project Website*, 11(2007):21.

[31] Brabham, D. C. (2012). A model for leveraging online communities. *The participatory cultures handbook*, 120.

[32] Braem, B., Blondia, C., Barz, C., Rogge, H., Freitag, F., Navarro, L., Bonicioli, J., Papathanasiou, S., Escrich, P., Baig Viñas, R., et al. (2013). A case for research with and on community networks. *ACM SIGCOMM Computer Communication Review*, 43(3):68–73.

[33] Bui, N., Castellani, A. P., Casari, P., and Zorzi, M. (2012). The internet of energy: a web-enabled smart grid system. *Network, IEEE*, 26(4):39–45.

[34] Cerdà-Alabern, L., Neumann, A., and Escrich", P. (2013). Experimental evaluation of a wireless community mesh network. In *The 16th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, MSWiM'13*. ACM.

[35] Chen, H., Li, Y., and Shi, W. (2012). Fine-grained power management using process-level profiling. *Sustainable Computing: Informatics and Systems*, 2(1):33–42.

[36] Choi, S., Kim, H., Byun, E., Baik, M., Kim, S., Park, C., and Hwang, C. (2007). Characterizing and classifying desktop grid. In *Cluster Computing and the Grid, 2007. CCGRID 2007. Seventh IEEE International Symposium on*, pages 743–748. IEEE.

[37] Chun, B.-G., Iannaccone, G., Iannaccone, G., Katz, R., Lee, G., and Niccolini, L. (2010). An energy case for hybrid datacenters. *ACM SIGOPS Operating Systems Review*, 44(1):76–80.

[38] Clos, C. (1953). A study of non-blocking switching networks. *Bell System Technical Journal*, 32(2):406–424.

[39] Colmant, M., Kurpicz, M., Felber, P., Huertas, L., Rouvoy, R., and Sobe, A. (2015). Process-level power estimation in vm-based systems. In *European Conference on Computer Systems (EuroSys)*, page 14. ACM.

[40] Coroama, V. and Hilty, L. M. (2009). Energy consumed vs. energy saved by ict–a closer look. In *Environmental Informatics and Industrial Environmental Protection: Concepts, Methods and Tools, 23rd International Conference on Informatics for Environmental Protection, Berlin*, pages 353–361.

[41] Costa, P., Donnelly, A., O'Shea, G., and Rowstron, A. (2013). Camcubeos: a key-based network stack for 3d torus cluster topologies. In *Proceedings of the 22nd international symposium on High-performance parallel and distributed computing*, pages 73–84. ACM.

[42] Dán, G., Bobba, R. B., Gross, G., and Campbell, R. H. (2013). Cloud computing for the power grid: From service composition to assured clouds. In *In Proceedings of 5th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud'13)*.

[43] Dilley, J., Maggs, B., Parikh, J., Prokop, H., Sitaraman, R., and Weihl, B. (2002). Globally distributed content delivery. *Internet Computing, IEEE*, 6(5):50–58.

[44] Duan, H., Chen, C., Min, G., and Wu, Y. (2016). Energy-aware scheduling of virtual machines in heterogeneous cloud computing systems. *Future Generation Computer Systems*.

[45] Escrich, P., Baig, R., Vilata, I., Neumann, A., Aymerich, M., Lopez, E., Vega, D., Meseguer, R., Freitag, F., and Navarro, L. (2013). Community home gateways for p2p clouds. In *Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on*, pages 1–2. IEEE.

[46] Fang, X. et al. (2011). Smart grid—the new and improved power grid: A survey. *IEEE Communications Surveys and Tutorials*, 14(4):1 – 37.

[47] Farrington, N., Porter, G., Radhakrishnan, S., Bazzaz, H. H., Subramanya, V., Fainman, Y., Papen, G., and Vahdat, A. (2011). Helios: a hybrid electrical/optical switch architecture for modular data centers. *ACM SIGCOMM Computer Communication Review*, 41(4):339–350.

[48] Feeney, L. M. and Nilsson, M. (2001). Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1548–1557. IEEE.

[49] Gao, P. X., Curtis, A. R., Wong, B., and Keshav, S. (2012). It's not easy being green. *ACM SIGCOMM Computer Communication Review*, 42(4):211–222.

[50] Garg, S. K., Versteeg, S., and Buyya, R. (2013). A framework for ranking of cloud computing services. *Future Generation Computer Systems*, 29(4):1012–1023.

[51] Giacobbe, M., Celesti, A., Fazio, M., Villari, M., and Puliafito, A. (2015). Towards energy management in cloud federation: A survey in the perspective of future sustainable and cost-saving strategies. *Computer Networks*, 91:438–452.

[52] Goodrich, M. T. (2011). Data-oblivious external-memory algorithms for the compaction, selection, and sorting of outsourced data. In *Proceedings of the twenty-third annual ACM symposium on Parallelism in algorithms and architectures*, pages 379–388. ACM.

[53] Greenberg, A., Hamilton, J. R., Jain, N., Kandula, S., Kim, C., Lahiri, P., Maltz, D. A., Patel, P., and Sengupta, S. (2009). Vl2: a scalable and flexible data center network. In *ACM SIGCOMM Computer Communication Review*, volume 39, pages 51–62. ACM.

[54] Group, I. W. (2014). Ipcc wk. group iii – mitigation of climate change, annex ii i: Technology - specific cost and performance parameters.

[55] Gulati, A., Holler, A., Ji, M., Shanmuganathan, G., Waldspurger, C., and Zhu, X. (2012). Vmware distributed resource management: Design, implementation, and lessons learned. *VMware Technical Journal*, 1(1):45–64.

[56] Guo, C., Lu, G., Li, D., Wu, H., Zhang, X., Shi, Y., Tian, C., Zhang, Y., and Lu, S. (2009). Bcube: a high performance, server-centric network architecture for modular data centers. *ACM SIGCOMM Computer Communication Review*, 39(4):63–74.

[57] Guo, C., Wu, H., Tan, K., Shi, L., Zhang, Y., and Lu, S. (2008). Dcell: a scalable and fault-tolerant network structure for data centers. *ACM SIGCOMM Computer Communication Review*, 38(4):75–86.

[58] Gyarmati, L. and Trinh, T. A. (2010). How can architecture help to reduce energy consumption in data center networking? In *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*, pages 183–186. ACM.

[59] Hähnel, M., Döbel, B., Völp, M., and Härtig, H. (2012). Measuring energy consumption for short code paths using rapl. *ACM SIGMETRICS Performance Evaluation Review*, 40(3):13–17.

[60] Hameed, A., Khoshkbarforoushha, A., Ranjan, R., Jayaraman, P. P., Kolodziej, J., Balaji, P., Zeadally, S., Malluhi, Q. M., Tziritas, N., Vishnu, A., et al. (2014). A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems. *Computing*, pages 1–24.

[61] Hammadi, A. and Mhamdi, L. (2014). A survey on architectures and energy efficiency in data center networks. *Computer Communications*, 40:1–21.

[62] Hao, S., Li, D., Halfond, W. G., and Govindan, R. (2013). Estimating mobile application energy consumption using program analysis. In *Software Engineering (ICSE), 2013 35th International Conference on*, pages 92–101. IEEE.

[63] Hefeeda, M. and Saleh, O. (2008). Traffic modeling and proportional partial caching for peer-to-peer systems. *Networking, IEEE/ACM Transactions on*, 16(6):1447–1460.

[64] Helsley, M. (2009). Lxc: Linux container tools. *IBM devloperWorks Technical Library*.

[65] Huang, L., Jia, Q., Wang, X., Yang, S., and Li, B. (2011). Pcube: Improving power efficiency in data center networks. In *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, pages 65–72. IEEE.

[66] Isci, C., McIntosh, S., Kephart, J., Das, R., Hanson, J., Piper, S., Wolford, R., Brey, T., Kantner, R., Ng, A., et al. (2013). Agile, efficient virtualization power management with low-latency server power states. In *ACM SIGARCH Computer Architecture News*, volume 41, pages 96–107. ACM.

[67] Jin, X., Zhang, F., Wang, L., Hu, S., Zhou, B., and Liu, Z. (2014). A joint optimization of operational cost and performance interference in cloud data centers. *arXiv preprint arXiv:1404.2842*.

[68] Kaewpuang, R., Chaisiri, S., Niyato, D., Lee, B., and Wang, P. (2013a). Cooperative virtual machine management in smart grid environment.

[69] Kaewpuang, R., Chaisiri, S., Niyato, D., Lee, B., and Wang, P. (2013b). Cooperative virtual machine management in smart grid environment. *IEEE Transaction on Services Computing*.

[70] Kalogeraki, V., Gunopulos, D., and Zeinalipour-Yazti, D. (2002). A local search mechanism for peer-to-peer networks. In *Proceedings of the eleventh international conference on Information and knowledge management*, pages 300–307. ACM.

[71] Kansal, A., Zhao, F., Liu, J., Kothari, N., and Bhattacharya, A. A. (2010). Virtual machine power metering and provisioning. In *Proceedings of the 1st ACM symposium on Cloud computing*, pages 39–50. ACM.

[72] Kermarrec, A.-M. and Van Steen, M. (2007). Gossiping in distributed systems. *ACM SIGOPS Operating Systems Review*, 41(5):2–7.

[73] Keshav, S. and Rosenberg, C. (2011). How internet concepts and technologies can help green and smarten the electrical grid. *ACM SIGCOMM Computer Communication Review*, 41(1):109–114.

[74] Khan, A. M., Sharifi, L., Veiga, L., and Navarro, L. (2013). Clouds of Small Things: Provisioning Infrastructure-as-a-Service from within Community Networks. In *2nd International Workshop on Community Networks and Bottom-up-Broadband (CNBuB 2013), within IEEE WiMob*, Lyon, France.

[75] Khosravi, A., Garg, S. K., and Buyya, R. (2013). Energy and carbon-efficient placement of virtual machines in distributed cloud data centers. In *Euro-Par 2013 Parallel Processing*, pages 317–328. Springer.

[76] Kim, N., Cho, J., and Seo, E. (2012). Energy-credit scheduler: An energy-aware virtual machine scheduler for cloud systems. *Future Generation Computer Systems*.

[77] Koller, R., Verma, A., and Neogi, A. (2010). Wattapp: an application aware power meter for shared data centers. In *Proceedings of the 7th international conference on Autonomic computing*, pages 31–40. ACM.

[78] Kong, F. and Liu, X. (2015). A survey on green-energy-aware power management for datacenters. *ACM Computing Surveys (CSUR)*, 47(2):30.

[79] Lee, Y. C. and Zomaya, A. Y. (2011). Energy conscious scheduling for distributed computing systems under different operating conditions. *Parallel and Distributed Systems, IEEE Transactions on*, 22(8):1374–1381.

[80] Lenk, A., Klems, M., Nimis, J., Tai, S., and Sandholm, T. (2009). What's inside the cloud? an architectural map of the cloud landscape. In *Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing*, pages 23–31. IEEE Computer Society.

[81] Lin, D., Liu, Y., Hamdi, M., and Muppala, J. (2012). Flatnet: Towards a flatter data center network. In *Global Communications Conference (GLOBECOM), 2012 IEEE*, pages 2499–2504. IEEE.

[82] Liu, Y., Xiao, L., Liu, X., Ni, L. M., and Zhang, X. (2005). Location awareness in unstructured peer-to-peer systems. *Parallel and Distributed Systems, IEEE Transactions on*, 16(2):163–174.

[83] Lo, D., Cheng, L., Govindaraju, R., Barroso, L. A., and Kozyrakis, C. (2014). Towards energy proportionality for large-scale latency-critical workloads. In *ACM SIGARCH Computer Architecture News*, volume 42, pages 301–312. IEEE Press.

[84] Lua, E. K., Crowcroft, J., Pias, M., Sharma, R., and Lim, S. (2005). A survey and comparison of peer-to-peer overlay network schemes. *IEEE Communications Surveys and Tutorials*, 7(1-4):72–93.

[85] M. Selimi, F. F. (2014). Towards application deployment in community network clouds. *ICCSA*.

[86] Maheshwari, K., Lim, M., Wang, L., Birman, K., and van Renesse, R. (2013). Toward a reliable, secure and fault tolerant smart grid state estimation in the cloud. *Innovative Smart Grid Technologies*.

[87] Makkes, M. X., Taal, A., Osseyran, A., and Grosso, P. (2013). A decision framework for placement of applications in clouds that minimizes their carbon footprint. *Journal of Cloud Computing*, 2(1):1–13.

[88] Mattern, F., Staake, T., and Weiss, M. (2010). Ict for green: how computers can help us to conserve energy. In *Proceedings of the 1st international conference on energy-efficient computing and networking*, pages 1–10. ACM.

[89] McCullough, J. C., Agarwal, Y., Chandrashekar, J., Kuppuswamy, S., Snoeren, A. C., and Gupta, R. K. (2011). Evaluating the effectiveness of model-based power characterization. In *USENIX Annual Technical Conf*, volume 20.

[90] Meisner, D., Gold, B. T., and Wenisch, T. F. (2009). Powernap: eliminating server idle power. In *ACM Sigplan Notices*, volume 44, pages 205–216. ACM.

[91] Mezmaz, M., Melab, N., Kessaci, Y., Lee, Y. C., Talbi, E.-G., Zomaya, A. Y., and Tuyttens, D. (2011). A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems. *Journal of Parallel and Distributed Computing*, 71(11):1497–1508.

[92] Mohsenian-Rad, H. and Leon-Garcia, A. (2010). Coordination of cloud computing and smart power grids. *First IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 368 – 372.

[93] Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. *Consulted*, 1(2012):28.

[94] Nedevschi, S., Ratnasamy, S., Padhye, J., and Reserch, I. (2008). Hot data centers vs. cool peers. In *HotPower*.

[95] Nesmachnow, S., Dorronsoro, B., Pecero, J. E., and Bouvry, P. (2013). Energy-aware scheduling on multicore heterogeneous grid computing systems. *Journal of grid computing*, 11(4):653–680.

[96] Nesmachnow, S., Perfumo, C., and Goiri, Í. (2015). Holistic multiobjective planning of datacenters powered by renewable energy. *Cluster Computing*, 18(4):1379–1397.

[97] NIST (2010). Nist framework and roadmap for smart grid interoperability standards. *National Institute of Standards and Technology*.

[98] Orgerie, A.-C., Assuncao, M. D. d., and Lefevre, L. (2014). A survey on techniques for improving the energy efficiency of large-scale distributed systems. *ACM Computing Surveys (CSUR)*, 46(4):47.

[99] Orgerie, A.-C., Lefevre, L., and Gelas, J.-P. (2010). Demystifying energy consumption in grids and clouds. In *Green Computing Conference, 2010 International*, pages 335–342. IEEE.

[100] Payberah, A. H., Kavalionak, H., Kumaresan, V., Montresor, A., and Haridi, S. (2012). Clive: Cloud-assisted p2p live streaming. In *Peer-to-Peer Computing (P2P), 2012 IEEE 12th International Conference on*, pages 79–90. IEEE.

[101] Peter Mell, T. G. (2011). The nist definition of cloud computing. *Computer Security Division Information Technology Laboratory National Institute of Standards and Technology Gaithersburg, MD 20899-8930*.

[102] Pinel, F., Dorronsoro, B., Pecero, J. E., Bouvry, P., and Khan, S. U. (2013). A two-phase heuristic for the energy-efficient scheduling of independent tasks on computational grids. *Cluster computing*, 16(3):421–433.

[103] Popek, G. J. and Goldberg, R. P. (1974). Formal requirements for virtualizable third generation architectures. *Communications of the ACM*, 17(7):412–421.

[104] Pore, M., Abbasi, Z., Gupta, S. K., and Varsamopoulos, G. (2012). Energy aware colocation of workload in data centers. In *High Performance Computing (HiPC), 2012 19th International Conference on*, pages 1–6. IEEE.

[105] Pouwelse, J., Langendoen, K., Lagendijk, R., and Sips, H. (2001). Power-aware video decoding. In *22nd Picture Coding Symposium, Seoul, Korea*, pages 303–306.

[106] Qian, Y.-H. and Zhou, Y. (1998). Complete galilean-invariant lattice bgk models for the navier-stokes equation. *EPL (Europhysics Letters)*, 42(4):359.

[107] Rethinagiri, S. K., Palomar, O., Moreno, J. A., Unsal, O., and Cristal, A. (2015). Vpm: Virtual power meter tool for low-power many-core/heterogeneous data center prototypes. In *Computer Design (ICCD), 2015 33rd IEEE International Conference on*, pages 651–658. IEEE.

[108] Rodrigues, P. S. D., da Cruz Ribeiro, C. N., and Veiga, L. (2010). Incentive mechanisms in peer-to-peer networks. In *15th IEEE Workshop on Dependable Parallel, Distributed and Network-Centric Systems (DPDNS)*.

[109] Roy, S., Rudra, A., and Verma, A. (2013). An energy complexity model for algorithms. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 283–304. ACM.

[110] Roy, S., Rudra, A., and Verma, A. (2014). Energy aware algorithmic engineering. In *Modelling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2014 IEEE 22nd International Symposium on*, pages 321–330. IEEE.

[111] Rusitschka, S., Eger, K., and Gerdes, C. (2010). Smart grid data cloud: A model for utilizing cloud computing in the smart grid domain. *First IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 483 – 488.

[112] Schubert, S., Kostic, D., Zwaenepoel, W., and Shin, K. G. (2012). Profiling software for energy consumption. In *Green Computing and Communications (GreenCom), 2012 IEEE International Conference on*, pages 515–522. IEEE.

[113] Sharifi, L., Cerda-Alabern, L., Freitag, F., and Veiga, L. (2016a). Energy Efficient cloud service provisioning: Keeping data center granularity in perspective. *Grid Computing journal*.

[114] Sharifi, L., Freitag, F., and Veiga, L. (2014a). Combing smart grid with community clouds: Next generation integrated service platform. In *Smart Grid Communications (SmartGridComm)*, pages 434–439. IEEE.

[115] Sharifi, L., Freitag, F., and Veiga, L. (2015). Envisioning cloud of energy. In *2015 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 229–234.

[116] Sharifi, L., Freitag, F., and Veiga, L. (2016b). Arta: An economic middleware to exchange pervasive energy and computing resources. In *Smart Grid Communications (SmartGridComm)*. IEEE.

[117] Sharifi, L., Rameshan, N., Freitag, F., and Veiga, L. (2014b). Energy efficiency dilemma: P2p-cloud vs. datacenter. In *Cloud Computing Technology and Science (CloudCom), 2014 IEEE 6th International Conference on*, pages 611–619. IEEE.

[118] Sharifi, L., Simao, J., Rameshan, N., Freitag, F., and Veiga, L. A framework to analyse energy effectiveness in p2p assisted cloud ecosystems. *Submitted to IEEE Transactions on Cloud Computing*.

[119] Shuja, J., Gani, A., Shamshirband, S., Ahmad, R. W., and Bilal, K. (2016). Sustainable cloud data centers: A survey of enabling techniques and technologies. *Renewable and Sustainable Energy Reviews*, 62:195–214.

[120] Sim, K. M. (2012). Agent-based cloud computing. *Services Computing, IEEE Transactions on*, 5(4):564–577.

[121] Simão, J. and Veiga, L. (2012). VM economics for java cloud computing - an adaptive and resource-aware java runtime with quality-of-Execution. In *The 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2012) — Doctoral Symposium: Cloud Scheduling, Clusters and Data Centers*. IEEE.

[122] Simão, J. and Veiga, L. (2013). Flexible SLAs in the cloud with partial utility-driven scheduling. In *IEEE 5th International Conference on Cloud Computing Technology and Science (CloudCom 2013)*. IEEE.

[123] Singh, A. K. (2012). Smart grid cloud. *International Journal of Engineering Research and Applications*, pages 674 – 704.

[124] Stoess, J., Lang, C., and Bellosa, F. (2007). Energy management for hypervisor-based virtual machines. In *USENIX annual technical conference*, pages 1–14.

[125] Subirats, J. and Guitart, J. (2014). Assessing and forecasting energy efficiency on cloud computing platforms. *Future Generation Computer Systems*.

[126] Tang, L., Mars, J., and Soffa, M. L. (2011). Contentiousness vs. sensitivity: improving contention aware runtime systems on multicore architectures. In *Proceedings of the 1st International Workshop on Adaptive Self-Tuning Computing Systems for the Exaflop Era*, pages 12–21. ACM.

[127] Tei, K. and Gurgen, L. (2014). Clout: Cloud of things for empowering the citizen clout in smart cities. In *Internet of Things (WF-IoT), 2014 IEEE World Forum on*, pages 369–370. IEEE.

[128] Tolia, N., Wang, Z., Marwah, M., Bash, C., Ranganathan, P., and Zhu, X. (2008). Delivering energy proportionality with non energy-proportional systems-optimizing the ensemble. *HotPower*, 8:2–2.

[129] Valancius, V., Laoutaris, N., Massoulié, L., Diot, C., and Rodriguez, P. (2009a). Greening the internet with nano data centers. In *Proceedings of the 5th*

international conference on Emerging networking experiments and technologies, pages 37–48. ACM.

[130] Valancius, V., Laoutaris, N., Massoulié, L., Diot, C., and Rodriguez, P. (2009b). Greening the internet with nano data centers. In *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*. ACM.

[131] Versick, D., Waßmann, I., and Tavangarian, D. (2013). Power consumption estimation of cpu and peripheral components in virtual machines. *ACM SIGAPP Applied Computing Review*, 13(3):17–25.

[132] Vytelingum, P., Ramchurn, S., Voice, T., Rogers, A., and Jennings, N. (2011). Agent-based modeling of smart-grid market operations. In *IEEE Power and Energy Society General Meeting*, pages 1–8.

[133] Wang, C., Nasiriani, N., Kesidis, G., Urgaonkar, B., Wang, Q., Chen, L. Y., Gupta, A., and Birke, R. (2015). Recouping energy costs from cloud tenants: Tenant demand response aware pricing design. In *ACM e-energy 2015*.

[134] Wang, G., Andersen, D. G., Kaminsky, M., Papagiannaki, K., Ng, T., Kozuch, M., and Ryan, M. (2010). c-through: Part-time optics in data centers. In *ACM SIGCOMM Computer Communication Review*, volume 40, pages 327–338. ACM.

[135] Wang, S., Chen, H., and Shi, W. (2011). Span: A software power analyzer for multicore computer systems. *Sustainable Computing: Informatics and Systems*, 1(1):23–34.

[136] Wang, S.-H., Huang, P.-W., Wen, C.-P., and Wang, L.-C. (2014). Eqvmp: Energy-efficient and qos-aware virtual machine placement for software defined datacenter networks. In *Information Networking (ICOIN), 2014 International Conference on*, pages 220–225. IEEE.

[137] Wong, D. and Annavaram, M. (2012). Knightshift: Scaling the energy proportionality wall through server-level heterogeneity. In *Microarchitecture (MICRO), 2012 45th Annual IEEE/ACM International Symposium on*, pages 119–130. IEEE.

[138] Xiao, L., Liu, Y., and Ni, L. M. (2005). Improving unstructured peer-to-peer systems by adaptive connection establishment. *Computers, IEEE Transactions on*, 54(9):1091–1103.

[139] Xiao, Z., Song, W., and Chen, Q. (2013). Dynamic resource allocation using virtual machines for cloud computing environment. *Parallel and Distributed Systems, IEEE Transactions on*, 24(6):1107–1117.

[140] Y., S. et al. (2010). On using cloud platforms in a software architecture for smart energy grids. *IEEE International Conference on Cloud Computing (CloudCom)*.

[141] Zhai, Y., Zhang, X., Eranian, S., Tang, L., and Mars, J. (2014). Happy: Hyperthread-aware power profiling dynamically. In *2014 USENIX Annual Technical Conference (USENIX ATC 14)(Philadelphia, PA*, pages 211–217.

[142] Zhang, G. and Zhang, G. (2007). Agent selection and p2p overlay construction using global locality knowledge. In *Networking, Sensing and Control, 2007 IEEE International Conference on*, pages 519–524. IEEE.

[143] Zhang, Q., Zhani, M. F., Zhang, S., Zhu, Q., Boutaba, R., and Hellerstein, J. L. (2012). Dynamic energy-aware capacity provisioning for cloud computing environments. In *Proceedings of the 9th international conference on Autonomic computing*, pages 145–154. ACM.

[144] Zhang, X., Li, Z., and Wang, Y. (2008). A distributed topology-aware overlays construction algorithm. In *Proceedings of the 15th ACM Mardi Gras conference*, page 5. ACM.

[145] Zhang, X. Y., Zhang, Q., Zhang, Z., Song, G., and Zhu, W. (2004). A construction of locality-aware overlay network: moverlay and its performance. *Selected Areas in Communications, IEEE Journal on*, 22(1):18–28.

[146] Zhi, J., Bila, N., and de Lara, E. (2016). Oasis: energy proportionality with hybrid server consolidation. In *Proceedings of the Eleventh European Conference on Computer Systems*, page 10. ACM.

[147] Zhiqun, X., Duan, C., Zhiyuan, H., and Qunying, S. (2013). Emerging of telco cloud. *Communications, China*, 10(6):79–85.